System Analysis & Design & Software Engineering

MBA IT-310

DIRECTORATE OF DISTANCE EDUCATION

SWAMI VIVEKANAND
SUBHARTI UNIVERSITY
Meerut (National Capital Region Delhi)

Contents

SYLLABUS

System Analysis & Design and Software Engineering

Course Code: MBA IT -310		
Course Credit: 3	**Lecture: 2**	**Tutorial-1**
Course Type:	Skill Enhancement Course	
Lectures delivered:	**20 L+10T**	

End Semester Examination System

Maximum Marks Allotted	Minimum Pass Marks	Time Allowed
70	28	3 Hours

Continuous Comprehensive Assessment (CCA) Pattern

Tests	Assignment/ Tutorial/ Presentation/class test	Attendance	Total
15	5	10	30

Course Objective: The aim of the course is to introduce students to:

- System Concepts and System Planning
- Tools of Structured Analysis
- Basics of Information Security

UNIT	Course Content	Hours
I	**Systems Concept:** Characteristics of a System; Elements of System; Types of System; Decision Support System; System Development LIfe Cycle, Investigation, Analysis, Design, Implementation, Post Implementation Review and Maintenance.	8
II	**Systems Planning and Investigation :** Basis for planning in Systems Analysis-Dimensions of Planning, Initial Investigation, Needs Identification, Determining the User's Information Requirements, Feasibility Study, Feasibility Considerations, Steps in Fesibility Analysis-Feasibility Report	8
III	**Tools of Structured Analysis:** Data Flow Diagram (DFD), Entity Relationship Diagrams, Data Dictionary, Process Modeling: Structured English, Decision Tree and Decision Table, Object Oriented Analysis (OOA) and Object Oriented Design (OOD)	6
IV	**Basics of Information Security :** Types of Attacks, Viruses, Virus control, Hackers, Overview of Risks associated with Internet, Itrusion Detection Risk managemnt, Disaster Recovery Plan, Cryptography and Authentication, Managing Risk, Information Security Policy, Creting a Secure Environment, Internet Security Standards	8

[illegible]

An Introduction to System Analysis and Design

Notes

Structure

1.0 LEARNING OBJECTIVES

After reading this chapter students will be able to:

- Know about concept of systems
- Discuss the types and elements of systems

Notes

- Know about the characteristics of systems
- Describe the Decision support system and development of life cycle
- Understand the analysis and design.
- Discuss the implementation review and maintenance

1.1 INTRODUCTION

The terrn 'system' is derived from the Greek word 'system' (to combine), which means an organized relationship among functioning units or components. A system exists because it is designed to achieve one or more objectives. A system is an orderly arrangement ofits components. The components of a system have structure and order. The organization determines the flow of control, communication and the chain of commands.

There are many system concepts which play an important role in understanding the system. The flow of information in an organization is very vital. There are various departments in an organization, depending on the services or products they provide to us. With each department there are three traditional levels of management-top, middle and lower. For making the proper decisions-the different levels of managers require the right kind. of information at right time. Information syatem is a system that provides information to people in an organization. There are variour types of computer-based information systems, which serve different levels tif :m anagement.

1.2 SYSTEM AND ITS PARTS

A system isan interrelated set of components with an identifiable boundary working together for some purpose. A system has nine characteristics (see Fig. 1).

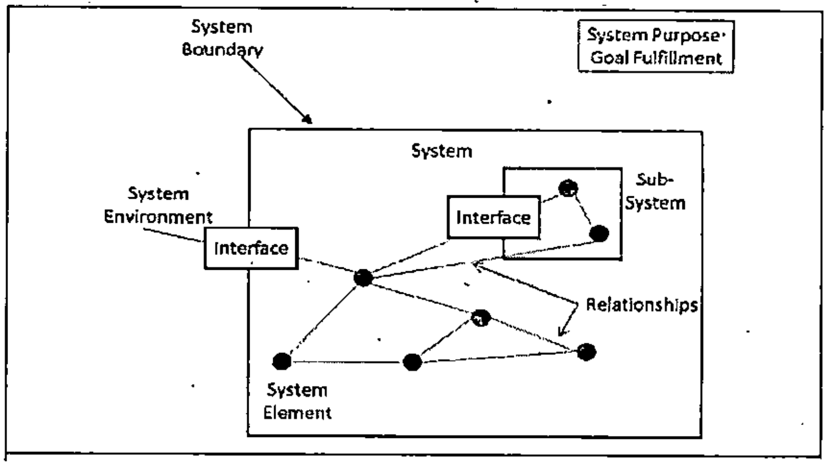


Fig. 1. A general illustration of a system.

(i) Components

(ii) Interrelated components

(iii) A boundary

(iv) A purpose

(u) An environment

(ui) Interfaces

(vii) Input

(viii) Output

(ix) Constraints

A system is made up of components. A component is either an irreducible .part or an aggregate of parts, also known as a subsystem. The simple concept of a component is very powerful. For example, just as with an automobile or a stereo system with proper design, we can repair or upgrade the system by changing individual components without having to make changes throughout the entire system.

The components are interrelated; that is, the function of one component is somehow tied to the functions of the other components. For example, the work of one component, such as producing a daily report of customer orders received, may not progress successfully until the work of another component is finished, such as sorting customer orders by date of their receipt.

A system has a boundary within which all of its components are contained and that establishes the limits of a system, separating the system from other systems. Components within the boundary of a system can be changed, whereas things outside the boundary cannot be changed. All of the components work together to achieve some overall purpose for the larger system: the system's main reason for existing.

A system exists within an environment, which comprises of everything outside the system's boundary. For example, we might consider the environment of a state university to include the legislature, prospective students, foundations and funding agencies, and the news media. Usually the system interacts with its environment, exchanging, in the case of an information system, data: and information. The points at which the system meets its environment are known as interfaces, and there are also interfaces between subsystems. An example of subsystem interface is the clutch subsystem, which acts as the point of interaction between the engine and transmission subsystems of a car.

Special characteristics of interfaces are given below:

Interface Functions

Because an interface exists at the point where a system meets its environment, the interface has several special, important functions. An interface provides

- Security, protecting the system from undesirable elements that may want to infiltrate it

Notes

- Filtering unwanted data, both the elements leaving the system and entering it
- Coding and decoding incoming and outgoing messages
- Detecting the correcting errors in its interaction with the environment
- Buffering, providing a layer of slack between the system and its environment, so that the system and its environment can work on different cycles and at different speeds
- Summarizing raw data and transforming them into the level of detail and format required throughout the system (for an input interface) or in the environment (for an output interface)

Because interface functions are critical in communication between system components or a system and its environment, interfaces receive much attention in the design of information systems.

It is the design of good interfaces that allows different systems to work together without being too dependent on each other.

A system must face constraints in its functioning because there are limits (in terms of capacity, speed, or capabilities) to what it can do and how it can achieve its purpose within its environment. Some of these constraints are imposed inside the system (e.g., a limited number of staff available), whereas others are imposed by the environment (e.g., due dates or regulations imposed by government or some other agency). A system takes input from its environment in order to function.

Mammals, for example, take in food, oxygen, and water from the environment as input. Finally, a system returns output to its environment as a result to its functioning and thus achieves its purpose.

Now you are familiar with the definition of a system and its nine important characteristics let us take an example of a system and use it to illustrate the definition and each system characteristic. Consider a system that is familiar to you: a fast-food restaurant (see Figure 1.2).

How is a fast-food restaurant a system?

Let us take a look at the fictional Roop Chand restaurant in New Delhi, India. First, it has components, or subsystems. The physical subsystems are: kitchen, dining room, counter, storage, and office. As you might expect the subsystems are interrelated and work together to prepare food and deliver it to customers, one purpose for the restaurant's existence. Food is delivered daily, kept in storage) prepared in the kitchen, sold at the eounter, and often eaten in the dining room.

The boundary is represented by its physical walls, and the primary purpose for the restaurant's existence is to make a profit for its owners, Aman and Vansh Dixit. The restaurant's environment consists of those external elements that interact with it, such as customers (many of whom come from nearby Delhi University), the local labor supply, food distributors (much of the .produce is grown locally); banks, and neighborhood fast-food competitors. It has one interface at the counter, where customers place orders, and another at the back do.or, where food and supplies are delivered.

Still another interface is the telephone managers use regularly to talk with bankers and food distributors. The restaurant faces several constraints. It is designed for the easy and cost-effective preparation of certain popular foods, such as hamburgers and coffees, which constrains the restaurant in the foods it may offer for sale. Its size and its location in the university neighborhood constrain how much money it can make on any given day.

The MCD Health Department also imposes constraints. such as rules governing food storage, Inputa include, but are not limited to, ingredients for the burgers and other food as well as cash and labor. Outputs include, but are not limited to, prepared food, bank deposits, and trash.

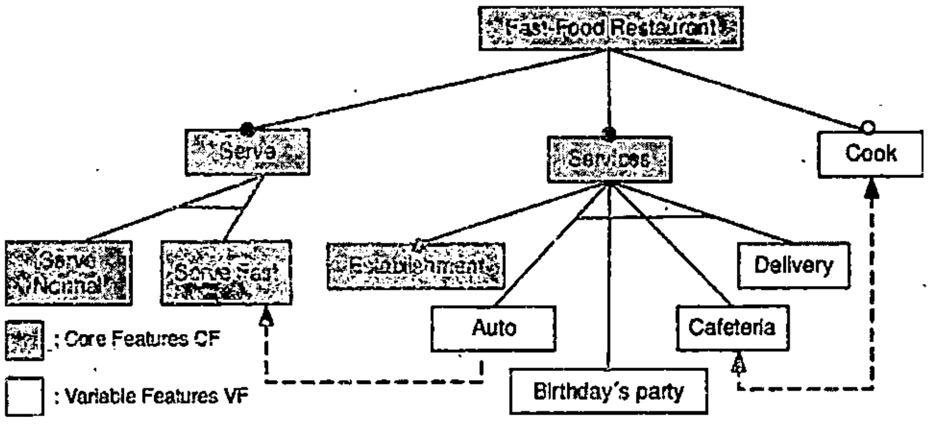


Fig. 2 A fast-food restaurant as a system.

1.3 BASIC CONCEPTS OF A SYSTEM

Once we have recognised something as a system and identified the system's characteristics, how do we understand the system? Further, what principles or concepts about systems help the design of information systems? A key aspect of a system is the system's relatfonship with its environment. Some systems, called open system, interact freely with their environments, taking ininput and returning output.

As the environment changes, an open system must adapt to the changes or suffer the consequences. A closed system does not interact with the environment; changes in the environment and adaptability are not issues for a closed system. However; all business information systems are open, and in order to.understand a system and its relationships to other inforn:uition systems, to the organization, and to the larger environment, you must always think of info.rmation systems as open and constantly interacting with the environment.

There are many other important systems concepts with which systems analysts (the key individuals in the systems development process) need to become familiar:

- Decomposition
- Coupling
- Modularity
- Cohesion

Notes

Decomposition

Decomposition deals with being able to break down a system into its components. These components may themselves be systems (subsystems) and can be broken down into their components as well. How does decomposition aid understanding of a system? Decomposition results in smaller and less complex pieces that are easier to understand than larger, complex pieces. Decomposing a system also helps us to focus on one particular part of a system, making it easier to think of how to modify that one part independently of the entire system.

Decomposition aids a systems analyst and other systems development project team members by:

- Breaking a system into smaller, more manageable, and understandable subsystems
- Facilitating the focusing of attention on one area (subsystem) at a time without interference from other parts
- Allowing attention to concentrate on the part of the system pertinent to a particular audience, without confusing people with details irrelevant to their interests
- Permitting different parts of the system to be built at independent times and/or by different person.

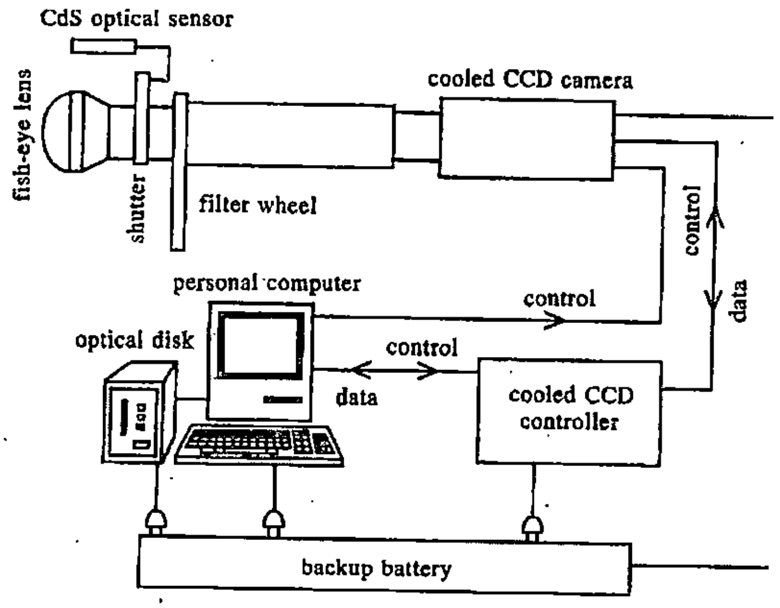


Fig. 3. An example of decomposition.

Figure 3 shows the decomposition of a portable compact disc (CD) player. At the highest level of abstraction, this system simply accepts CDs and settings of the volume and tone controls as input and produces music as output. Decomposing the system into subsystems provides the system's inner workings: There are separate systems for reading the digital signals from the CDs, for amplifying the signals, for turning the signals into sound waves, and for controlling the volume and tone of the sound. Breaking the subsystems down into their components would provide even more about the inner workings of the system and greatly enhance our understanding of how the overall works.

Modularity

Modularity a direct result of decomposition, means dividing a system up into chunks or modules of a relatively uniform size. Modules can represent a system simply, making it not only easier to understand, but also easier to redesign and rebuild.

Modularity is a frequently used term in information technology and computer science. Modularity refers to the concept of making multiple modules first and then linking and combining them to form a complete system. Modularity enables re-usability and minimizes duplication.

In addition to re-usability, modularity also makes it easier to fix problems as bugs can be traced to specific system modules, thus limiting the scope of detailed error searching. Modular programming is an extensively used concept based on modularity. Modularity is also a feature of object oriented programming.

We can define modularity as:

> Modularity is the degree to which a system's components are made up of relatively independent components or parts which can be combined.

Coupling

Coupling is the extent to which subsystems are dependent on each other. Subsystems should be as independent as possible. If one subsystem fails and other subsystems are highly dependent on it, the others will either fail themselves or have problems functioning, After looking at Figure 1.3, we would say the components of a portable CD player are tightly coupled. The amplifier and the unit that reads the CD signals are wired together in the same container, and the boundaries between these two subsystems may be difficult to draw clearly. If one subsystem fails, the entire CD player must be sent off for repair.

In a home stereo system, the components are loosely coupled because the subsystems, such as the speakers, the amplifier, the receiver, and the CD player, are all physically separate and function independently. For example, if the amplifier in a home stereo system fails, only the amplifier needs to be repaired.

Cohesion

Finally, cohesion is the extent to which a subsystem performs a single function. In biological systems, subsystems tend to be well differentiated, and thus very cohesive. In human made systems, subsystems are not always as cohesive as they should be.

One final key systems concept with which you should be familiar is the difference between logical and physical systems. Any description of a system is abstract because the definition is not the system itself. When we talk about logical and physical systems, we are actually talking about logical and physical system descriptions.

Notes

A logical system description gives the purpose and function of the system without tying the description to any specific physical implementation.

For example, in developing a logical description of the portable CD player, we describe the basic components of the player (signal reader, amplifier, speakers, controls and their relations to each other, focusing on the function of playing CDs using a self-contained, portable unit.

We do not specify whether the earphone jack contains aluminium or gold, where we could buy the laser that reads the CDs, or how much the jack or the laser cost to produce

On the other hand, the physical system description is a material depiction of the system, a central concern of which is building the system. A physical description of the portable CD player would provide details on the construction of each subunit, such as the design of the laser, the composition of the earphones, whether the controls feature digital readouts. A systems analyst should deal with function (logical system description) before form (physical system description), just as an architect does for the analysis and design of buildings before actual construction.

1.4 CHARACTERISTICS OF SYSTEM

A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective. The study of system concepts has three basic implications:

- A system must be designed to achieve a predetermined objective.
- Interrelationships and interdependence must exist among the components.
- The objectives of the organization as a whole have a higher priority than the objectives of its subsystems.

Characteristics of a system:

- **Organization:** It implies structure and order. It is the arrangement of components that helps to achieve objectives.
- **Interaction:** It refers to the manner in which each component functions with other components of the system.
- **Interdependence:** It means that parts of the organization or computer system depend on one another. They are coordinated and linked together according to a plan. One subsystem depends on the output of another subsystem for proper functioning.
- **Integration:** It refers to the holism of systems. It is concerned with how a system is tied together.

Central Objective:

A system should have a central objective. Objectives may be real or stated. Although a stated objective may be the real objective, it is not uncommon for an organization to state one objective and operate to achieve another. The important point is that users must know the central objective of a computer application early in the analysis for a successful design and conversion.

1.5 ELEMENTS OF SYSTEM

A system has three basic elements input, processing and output. The other elements include control, feedback, boundaries, environment and interfaces.

- **Input and output:** Input is what data the system receives to produce a certain output. What goes out from the system after being processed is known as output. Inputs are the information or elements that we enter the system for processing. Output is the outcome of processing. A major objective of a system is to produce an output that has value to its user. Whatever the nature of the output it must be in line with the expectations of the intended user. A system feeds on input to produce output.
- **Processing:** The process involved to transform input into output is known as Processing. It is the element of a system that involves the actual transformations of input into output. it is the operational component of a system. Processors may modify the input totally or partially, depending on the specifications of the output.
- **Control:** In order to get the desired results it is essential to monitor and control the input, Processing and the output of the system. This job is done by the control. The control element guides the system. It is the decision-making subsystem that controls the pattern of activities governing input, processing and output. In an organization context, management as a decision making body controls the inflow, handling and outflow of activities that affect the welfare of the business. In a computer system, the operating system and accompanying software influence the behavior of the system. Output specifications determine what and how much input is needed to keep the system in balance.
- **Feedback:** The Output is checked with the desired standards of the output set and the necessary steps are taken for achieving the output as per the standards, this process is called as Feedback. It helps to achieve a much better control in the system. Control in a dynamic system is achieved by feedback. Feedback measures output against a standard procedure that includes communication and control. After the output is compared against performance standards, changes can result in the input or processing and consequently the output. Feedback can be positive or negative. Positive feedback reinforces the performance of the system. Negative feedback provides some information for action that will help us to improve the quality of the output.
- **Boundaries:** The boundaries are nothing but the limit of the system. Setting up boundaries helps for better concentration of the actives carried in the system.
- **Environment:** The environment is the area where the organization operates. It is the source of external elements for a system. It often determines how a system must function. An environment may consist of vendors, competitors etc The things outside the boundary of the system are known as environment. Change in the environment affects the working of the system.
- **Interfaces:** The interconnections and the interactions between the sub-systems is known as the Interfaces. They may be inputs and outputs of the systems.

Notes

1.6 TYPES OF SYSTEMS

The frame of .reference within which.one views a system is related to the use of the systems approach for analysis. Systems have been classified in different ways. Common classifications are:

1. Physical or abstract,
2. Open or closed, and
3. "Man-made" information systems

Physical or Abstract systems

Physical systems are tangible entities that may be static or dynamic in operation. For example, the physical parts of the computer center are the offices, desks, and chairs that facilitate operation of the computer. They can be seen and counted; they are static. In contrast, a programmed computer is a dynamic system. Data, programs, output, and applications change as the user's demands or the priority of the information requested changes.

Abstract systems are conceptual or nonphysical entities. They may be as straightforward as formulas of relationships among sets of variables or models-the abstract conceptualization of physical situations. A model is a representation of a real or a planned system. The use of models makes it easier for the analyst to visualize relationships in the system under study. The objective is to point out the significant elements and the key interrelationships of a complex system.

System Models

In no field are models used more widely and with greater variety than in system analysis. The analyst begins by creating a model ofthe reality (facts, relationships, procedures, etc.) with which the system is concerned. Every computer system deals with the real world, a problem area, or a reality outside itself.

For example, a telephone switching system is made up of subscribers, telephone handsets, diitling, conference calls, and the like. The analyst begins by modeling this reality before considering the functions that the system is to perform

Various business system models are used to show the benefits of abstracting complex systems to model form. The major models discussed here are schematic, flow, static, and dynamic system models.

Schematic Models

A schematic model is a two-dimensional chart depicting system elements and their linkages. Fugure 4 shows the major elements of a personnel information system together with material and information flow.

Flow System Model.

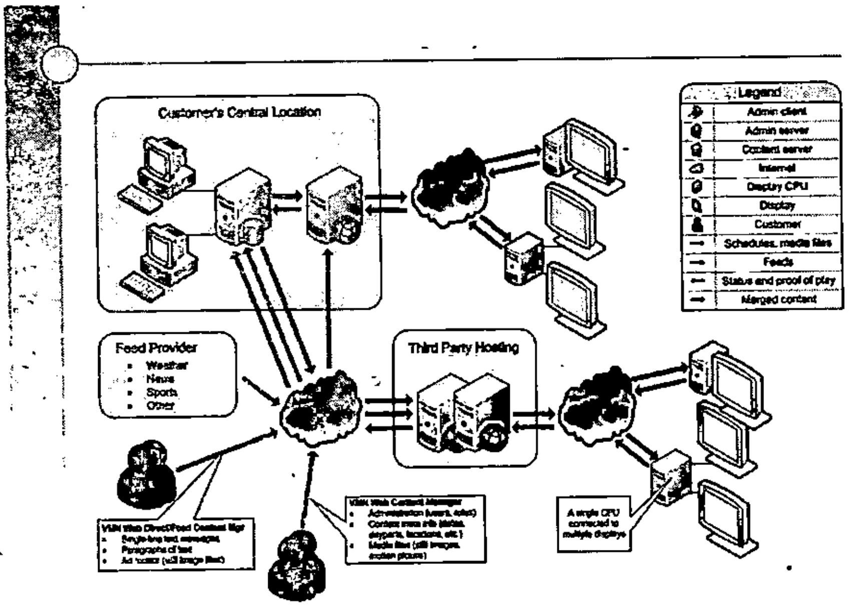


Fig. 4 Personnel information flow in a banking environment.

A flow system model shows the flow of the material, energy, and information that hold the system together. There is an orderly flow of logic in such models. A widely known example is PERT (Program Evaluation and Review Technique).

It is used to abstract a realworld system in model form, manipulate specific values to determine the critcal path, interpret the relationships, and relay them back as a control. The probability of completion within a time period is considered in connection with time, resources, and performance specifications (See Figure 5).

Static System Models

This type of model exhibits one pair of relationships such as activity time or cost quantity. The Gantt chart, for example, gives a static picture of an activity.;time telationship, In Figure 1.6 planned activities (stamping, sanding, etc,) are plotted in relation to time.

The date column has light lines that indicate the amount of time it takes to complete a given activity. The heavy line represents the comulative time schedule for each activity.

The stamping department for example, is scheduled to start working on order number 25 Wednesday mo!'ning, and complete the job by the same evening. One day is also scheduled for order number 28, two days for order number 22, and two days (August 10-11) for order number 29.

The total of six days is represented by the heavy line opposite the stamping .iepartment. The broken line indicates that the department is two days behind schedule. The arrowhead indicates the data when the chart is to be in effect.

Notes

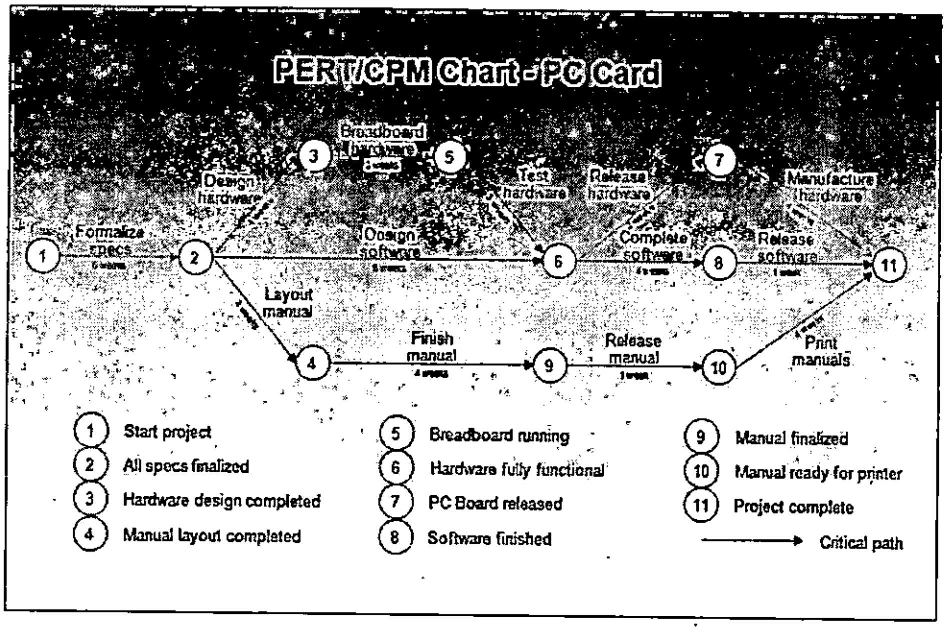


Fig. 5 A CPM chart showing flow system model

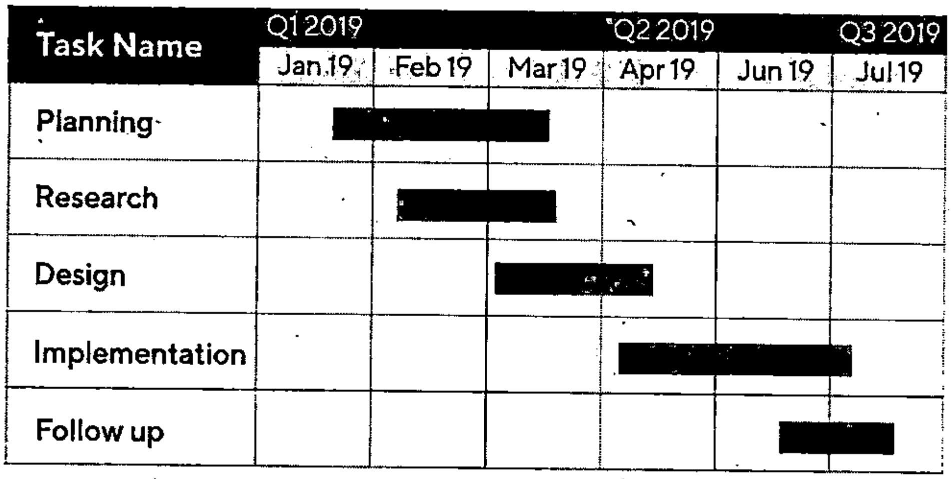


Fig. 6 Gantt chart-An example.

Dynamic System Models

Business organizations are dynamic systems. A dynamie model approximates the type of organization or applications that analysts deal with. It depicts an ongoing, constantly changing system. As mentioned earlier; it consists of (1) inputs that enter the system, (2) the processor through which transformation takes place, (3) the program(s) required for p_rocessing, and (4) the output(s) that result from processing.

Open or Closed Systems

Another classification of systems is based on their degree of independence An open system has many interfaces with its environment. It permits interaction across its boundary; it receives inputs from and delivers outputs to the outside. An information system falls into this category, since it must adapt to the changing demands of the user. In contrast, a cloBed system is isolated from environmental influeuces. In reality, a completely closed system is rare. In systems analysis, organizations, applications, and computers are invariably open, dynamic systems influenced by their environment.

A focus on the characteristics of an open system is particularly timely in the light of present-day business concerns with computer fraud, invasion of privacy, security controls, and ethics in computing. Whereas the technical aspects ofsystems analysis deal with internal routines within the user's application area, systems' analysis as an open system lends to expand the scope of analysis to relationships between the user area and other users and to environmental factors that must be considered before a new system is finally approved.

Furthermore, being open to suggestions implies that the analyst has to be flexible and the system being designed has to be responsive to the changing needs o.f the user and the environment.

Five important characteristics of open systems can be identified. These are given below:

1. *Input from Outside*

Open systems are self-adjusting and self regulating. When functioning properly, an open system reaches a steady state or equilibrium. In a retail firm, for example, a steady state exists when goods are purchased and sold without being either out of stock or over-stocked. An increase in the cost of goods forces a comparable increase in prices or 'decrease in operating cos.ts. This response .gives the firm its steady state.

2. *Entropy*

All dynamic systems tend to run down over time, resulting in entropy or loss of energy. Open systems resist entropy by seeking new inputs or modifying the processes to return to a steady state. In our example/no reaction to increase in cost of merchandise makes the business unprofitable which could force it into insolvency-a state of disorganization.

3. *Process, Output and Cycles*

Open systems produce useful output and operate in cycles, following a continuous flow path.

4. *Differentiation*

Open systems have a tendency toward an increasing speciaiization of functions and a greater differentiation of their components. In business, the roles of people and machines tend toward greater specialization and greater interaction. This characterililtie offers a compelling reason for the increasing value of the concept of systems in the systems, analyst's thinking.

Notes

5. Equifinality

The term implies that goals are achieved through differing courses of action and a variety of paths. In most systems, there is more of a consensus on goals than on paths to reach the goals. Understanding system characteristics helps analysts to identify their role and relate th.eir activities to the attainment of the firm's objectives as they undertake a system project. Analysts are themselves part of the organization They have opportunities to adapt the organization to changes through computerized applications so that the system does not "sun down". A key to this process is information feedback from the prime user of the new system as well as from top management.

Man-Made Information Systems

An information system is an open system that allows inputs and facilitates interaction with the user. These are discussed in detail later on in this unit.

1.7 DECISION SUPPORT SYSTEM

Decision support systems (DSS) are interactive software-based systems intended to help managers in decision-making by accessing large volumes of information generated from various related information systems involved in organizational business processes, such as office automation system, transaction processing system, etc.

DSS uses the summary information, exceptions, patterns, and trends using the analytical models. A decision support system helps in decision-making but does not necessarily give a decision itself. The decision makers compile useful information from raw data, documents, personal knowledge, and/or business models to identify and solve problems and make decisions.

Hence a decision support system (DSS) is a computer-based information system that provides a fiexible tool for analysis and helps managers focus on. the future. Whereas a TPS records data and an MIS summarizes data, a DSS analyzes data. To reach the DSS level of sophistication in information technology, an organization must have established TPS and MIS systems first. Some features of a DSS are given below :

Inputs and outputs. Inputs include internal data-such as summarized reports and processed transaction data-and also data that is external to the organization. External data may be produced by trade associations, marketing research firms, the Indian Bureau of the Census, and other government agencies.

The outputs are demand-reports on which a top manager can make decisions about unstructured problems.

Mainly for middle managers. A DSS is intended principally to assist middle managers in making tactical decisions. Questions addressed by the DSS might be, for example, whether interest rates will rise or whether there will be a strike in an important materials-supplying industry.

Produces analytic models. The key attribute of a DSS is that it uses models. A model is a mathematical representation of a real system. The models use a DSS database, which draws on the TPS and MIS files, as well as external data such as stock reports, government reports, and national and international news. The system is accessed using the DSS software.

The model allows the manager to do a simulation-play a "what-if' game-to reach decisions. Thus, the manager can simulate an aspect of the organization's environment in order to decide how to react to a change in conditions affecting it. By changing the hypothetical inputs to the model, the manager can see how the model's outputs are affected by doing so.

Many DSSs are developed to support the types of decisions faced by managers in specific industries, such as airlines or real estate. Curious how airlines decide how many seats to sell on a flight when so many passengers are no-shows? Wonder how owners of those big apartment complexes set rents and lease terms?

Investors in commercial real estate use a DSS called RealPlan to forecast property values up to 40 years into the future, based on income, expense, and cash-flow projections. Ever speculate aout how insurance carriers set different rates. Many companies use DSSs called geographic information systems (GISs), such as Mapinfo and Atlas GIS, which integrate geographic databases with other business data and display maps.

Programmed and Non-programmed Decisions

There are two types of decisions - programmed and non-programmed decisions.
Programmed decisions are basically automated processes, general routine work, where –

- These decisions have been taken several times.
- These decisions follow some guidelines or rules.
- For example, selecting a reorder level for inventories, is a programmed decision.

Non-programmed decisions occur in unusual and non-addressed situations, so –

It would be a new decision.

- There will not be any rules to follow.
- These decisions are made based on the available information.
- These decisions are based on the manger's discretion, instinct, perception and judgment.

For example, investing in a new technology is a non-programmed decision. Decision support systems generally involve non-programmed decisions. Therefore, there will be no exact report, content, or format for these systems. Reports are generated on the fly.

Attributes of a DSS

- Adaptability and flexibility
- High level of Interactivity
- Ease of use
- Efficiency and effectiveness
- Complete control by decision-makers
- Ease of development

Notes

- Extendibility
- Support for modeling and analysis
- Support for data access
- Standalone, integrated, and Web-based

Characteristics of a DSS

- Support for decision-makers in semi-structured and unstructured problems.
- Support for managers at various managerial levels, ranging from top executive to line managers.
- Support for individuals and groups. Less structured problems often requires the involvement of several individuals from different departments and organization level.
- Support for interdependent or sequential decisions.
- Support for intelligence, design, choice, and implementation.
- Support for variety of decision processes and styles.
- DSSs are adaptive over time.

Benefits of DSS

- Improves efficiency and speed of decision-making activities.
- Increases the control, competitiveness and capability of futuristic decision-making of the organization.
- Facilitates interpersonal communication.
- Encourages learning or training.
- Since it is mostly used in non-programmed decisions, it reveals new approaches and sets up new evidences for an unusual decision.
- Helps automate managerial processes.

Components of a DSS

Following are the components of the Decision Support System:

- Database Management System (DBMS) – To solve a problem the necessary data may come from internal or external database. In an organization, internal data are generated by a system such as TPS and MIS. External data come from a variety of sources such as newspapers, online data services, databases (financial, marketing, human resources).
- Model Management System – It stores and accesses models that managers use to make decisions. Such models are used for designing manufacturing facility, analyzing the financial health of an organization, forecasting demand of a product or service, etc.
- Support Tools – Support tools like online help; pulls down menus, user interfaces, graphical analysis, error correction mechanism, facilitates the user interactions with the system.

Classification of DSS

There are several ways to classify DSS. Hoi Apple and Whinstone classifies DSS as follows:

- Text Oriented DSS – It contains textually represented information that could have a bearing on decision. It allows documents to be electronically created, revised and viewed as needed.
- Database Oriented DSS – Database plays a major role here; it contains organized and highly structured data.
- Spreadsheet Oriented DSS – It contains information in spread sheets that allows create, view, modify procedural knowledge and also instructs the system to execute self-contained instructions. The most popular tool is Excel and Lotus 1-2-3.
- Solver Oriented DSS – It is based on a solver, which is an algorithm or procedure written for performing certain calculations and particular program type.
- Rules Oriented DSS – It follows certain procedures adopted as rules.
- Rules Oriented DSS – Procedures are adopted in rules oriented DSS. Export system is the example.
- Compound DSS – It is built by using two or more of the five structures explained above.

Types of DSS

Following are some typical DSSs:

- ***Status Inquiry System:*** It helps in taking operational, management level, or middle level management decisions, for example daily schedules of jobs to machines or machines to operators.
- ***Data Analysis System:*** It needs comparative analysis and makes use of formula or an algorithm, for example cash flow analysis, inventory analysis etc.
- ***Information Analysis System:*** In this system data is analyzed and the information report is generated. For example, sales analysis, accounts receivable systems, market analysis etc.
- ***Accounting System:*** It keeps track of accounting and finance related information, for example, final account, accounts receivables, accounts payables, etc. that keep track of the major aspects of the business.
- ***Model Based System:*** Simulation models or optimization models used for decision-making are used infrequently and creates general guidelines for operation or management.

1.8 ORGANIZATIONS, MANAGERS AND INFORMATION

At the heart of an organization is information and how it is used. To understand how to bring about change inan organization, we need to understand how organizations and their managers work-how they need, organize, and use information.

Notes

The flow of Information within an Organization. Take any sizable organization you are familiar with. Its main purpose is to perform a service or deliver a product. If it is nonprofit, for example, it may deliver service of educating students about AIDS or product of food for famine victims. If it is profit-oriented, it may, for example, sell the service of fixing computers or the product of eompu.ters themselves.

Information-whether computer-based or not-has to flow within an organization in a way that will help managers, and the organization, achieve their goals. At this end, organizations are often structured horizontally and vertically-horizontally to reflect functions and vertically to reflect management levels.

Departments

Depending on the services or products they provide, most organization have departments that perform five functions:

(i) **Research and Development (R & D):** Research and development. The research and development (R & DJ department does two things:

(1) It conducts basic research, relating discoyries to the organization's existing or new products.

(2) It does product development and tests and modifies new products or services created by researchers. Special software programs are available to aid in these functions.

(ii) **Production.** The production department produces the product or provides the service. In a manufacturing company, it takes the raw materials and has people or machinery turn them into finished goods. In many cases, this department uses CAD/CAM software and workstations, as well as robotics. In another type of company, this department might manage the purchasing, handle the inventories, and control the flow of goods and services.

(iii) **Marketing.** The marketing department looks after advertising, promotion, and sales. The people in this department plan, price, advertise, promote, package, and distribute the services or goods to customers' or clients. The sales representatives may use laptop computers, cellphones, wireless e mail, and faxes in their work while in their fields

(iv) **Accounting and finance.** The accounting and finance department handles all financial matters. It handles cash management, pays bills and taxes, issues paychecks, records payments, makes investments, and compiles financial statements and reports. It also produces financial budgets and forecasts financial performance after receiving information from other departments at certain time intervals.

(v) **Human resources.** The human resources, or personnel, department finds and hires people and adrrinisters sick leave and retirement matters. It is also concerned with compensation levels.. professional development, employee relations, and government regulations .

Management Levels

Within each of the five departments discussed above there are three traditional levels of management-top, middle, and lower. These levels are reflected .in the organization chart shown in Figure 7. An organization chart is a schematic drawing showing the hierarchy of formal relationships among an organization's employees. Managers on each of the three levels have different levels of responsibility and are therefore required to make different kinds of decisions. (See Figure 7).

- **Top managers.** The chief executive officer (CEO) or president is the very top manager. However, for our purposes, "top management" refers to the vice presidents, one of whom heads each department. Typical titles found at the top management level are treasurer, director, controller (chief accounting officer), and senior partner.

  Top managers are concerned with long-range, or strategic, planning and decisions. Strategic decisions are complex decisio.ns rarely based on predetermined routine procedures; they involve the subjective judgment of the decision maker. For example, strategic decisions might relate to how growth should be fina.nced and what new markets should be tackled first. Determining the company's 10-year goals, evaluating future financial resources, and formulating a response to competitors' actions are also strategic decisions.

  An AT & T vice president of marketing might have to make strategic decisions about promotional cam paigns to sell a new cable-modem service. The top manager who runs an electronics store might have to make strategic decisions about stocking a new line of personal digital assistants (PDAs)

- **Middle managers.** Some example of middle managers are plant manager, division manager, sales manager, branch manager, and director of personnel. Middle-level managers make tactical decisions to implement the strategic goals of the organization. A tactical decision is made without a base of clearly defined informational procedures; it may require detailed analysis and computations . An example might be deciding how many units of a specific product (PDAs, say) should be kept in inventory. Another is whether or not to purchase a larger computer system. The director of sales, who reports to the vice president of marketing for AT & T, sets sales targets for district sales managers throughout the country. They in turn feed him or her weekly and monthly sales reports.

- **Supervisory managers.** An example of a supervisory manager is a warehouse manager in charge of inventory restocking. Supervisory managers make operational decisions-predictable decisions that can be made by following well-defined sets of routine proced ures. These managers focus principally on supervising nonmanagement employees, monitoring day-to-day events, and taking corrective action where necessary.

  Determining not to restock inventory is an operation decision. (The guideline on when to restock may be determined at the level above.) A district sales manager for AT & T would monitor the promised sales and orders for cable modems coming in from sales representatives. When sales begin to drop off, the supervisor would need to take immediate action.

Notes

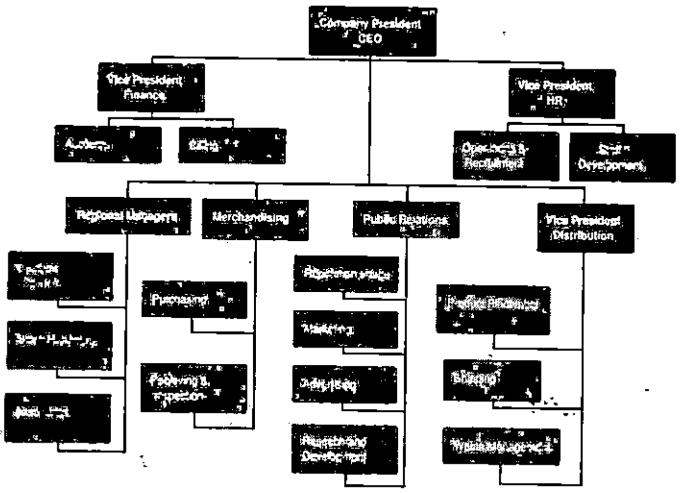


Fig. 7 An organization chart, and management levels and rest capabilities.

Types of Information

To make the propet decisions-strategic, tactical, operational-the different levels of managers need the right kind of information: structured, semistructured, and unstructured.

In general, all information to support intelligent decision making at all three levels must be correct-that is, accurate. It must also be complete, including all relevant data, yet concise, including only relevant data. It must be cost effective, meaning efficiently obtained, yet understandable. It must be current, meaning timely, yet also time sensitive, based on historical, current, or future information needs. This shows that information has three distinct properties as given below:

(i) Level of summarization

(ii) Degree of accuracy

(iii) Timeliness

These properties will be different for structured and unstructured information. Whether structured or unstructured information is more appropriate depends on the level of management and the type of decision making required. Structured information is detailed, current, not subjective, concerned with past events, records a narrow range of facts and covers an organization's internal activities.

Unstructured information is the opposite. Unstructured information is summarized, less current, highly subjective, concerned with future events, records a broad range of facts, and covers activities outside as well as inside an organization. Semi-structured information includes some structured information and some unstructured information.

As we have covered some basic concepts about how organizations are structured and what kinds of information are required at different levels of management, we need to examine what types of management information systems provide the information.

1.9 COMPUTER-BASED INFORMATION SYSTEMS

The main purpose of a computer-based information system is to provide managers (and various categories of employees) with the appropriate kind of information to help them make decisions. There are six types of ,omputer-based information systems which serve different levels of management. (See Figure 1.8).

- For lower managers. Transaction processing systems (TPSs)
- For middle managers. Management information systems (MISs) and decision support systems (DSSs)
- For top managers. Executive support systems (ESSs)
- For all levels, including nonmanagement. Office automation systems (OASs) and expert systems (ESs

Let us describe these

Transaction Processing Systems (TPSs)

In most organizations, particularly business organizations, most of what goes on consists largely of transactions. A transaction is a recorded event having to do with routine business activities. This includes everything concerning the product or service in which the organization is involved: production, distribution, sales, orders. It also includes materials purchased, employees hired, taxes paid etc.

These days in most organizations, the bulk of such transactions are recorded in a computer based information system. These systems tend to have clearly defined inputs and outputs, and.there is an emphasis on efficiency and accuracy. Transaction processing systems record data but do little in the way of converting data into information.

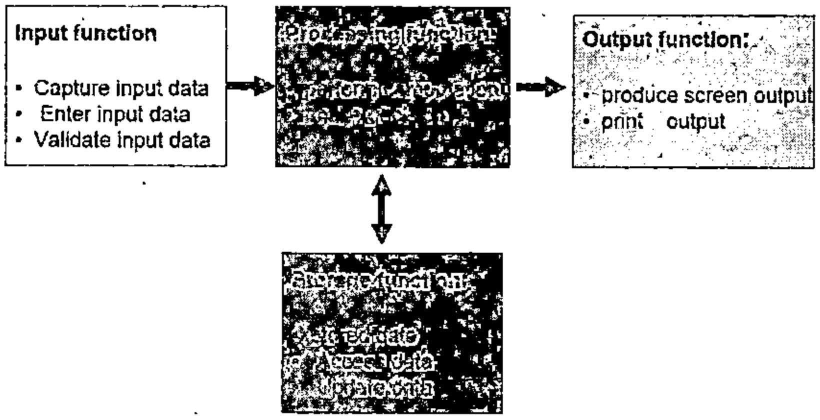


Fig. 8 Illustration of transaction processing system.

A transaction processing system (TPS) is a computer-based information system that keeps track of the transactions needed to cond uct business. Some features of a TPS are given below:

Notes

- **Input and output.** The inputs to the system are transaction data: bills, orders, inventory levels etc. The output consists of processed transactions bills, paychecks etc.
- **For lower managers.** Because the TPS deals with day-to-day matters, it' is principally of use to supervisory managers. That is, the TPS helps in making tactical decisions. Such systems are not usually helpful to middle or top managers in an organization.
- **Produces detail reports.** A manager at this level typically will receive information in the form of detail .rports. A detail report contains specific information about routine activiti - For example, the information .needed to decide whether to restock inventory.
- **Qne TPS for each department.** Each department or functional area of an organization-research and development, production, marketing, accounting and finance, and human resources'-usually has its own TPS. For example, the accounting and finance TPS handles, order pro.cessing, accounts receivable, inyentory and purchasing, accounts payable, order processing, and payroll.
- **Basis for MIS and DS.** The database of transactions stored in a TPS provides the basis for management information systems and decision support systems, as described next.

Management Information Systems (MISs)

A management information system (MIS) is a computer based information system that uses data recorded by TPS as input into programs that produce routine reports as output.

Feature of an MIS are given below:

- Input and output. Inputs consist of proc.essed transaction data, such as bills, orders, and paychecks, plus other internal data. Outputs consist of summarized, structured reports: budget summaries, production schedules etc.
- For middle managers. An MIS is intended principally to assist middle managers-specifically to help them with tactical decisions. It helps them to spot trends and get an overview of current business activities.
- Draws from all departments. The MIS draws from all five departments or functional 13:reas, not just one.
- Produces several kinds of reports. Managers at this level usuitlly receive information in the form of several kinds of reports: summary, exception, periodic , demand .
    1. *Summary* reports show totals and trends. For example, a report showing total sales by office, by product, and by salesperson, as well as total overall sales.
    2. *Exception* reports show out-of-the-ordinary data. For example, an inventory report listing only those items of which fewer than 20 are in stock.
    3. *Periodic reports* are produced on a regular schedule. Such daily, weekly, monthly, quarterly, or annual reports may have sales figures, income statements, or balance sheets. They are usually produced on paper, such as computer printouts.
    4. *Demand reports* produce information in response to an unscheduled demand. A director of finance might order a demand credit-background report on an unknown customer who wants to place a large order. Demand reports are often produced on a terminal or microcomputer screen, rather than on paper.

Executive Support Systems (ESSs)

An executive support system (ESS) is an easy-to-use DBS made especially for top managers; it specifically supports strategic decision making. An ESS is also known as executive information system (EIS). It draws on data not only from systems internal to the organization but also from those outside, such as news services or market-research databases. (See Figure 9).

An ESS might allow senior executives to call up predefined reports from their personal computers, whether desktops or laptops. They might, for instance, call up sales figures in many forms-by region, by week, by anticipated year, by projected increases. The ESS includes capabilities for analyzing data and doing "what-if' scenarios. ESSs also have the capability to browse through summarized information on all aspects of the organization and then zero in on ("drill down" to) detailed areas the manager believes require attention.

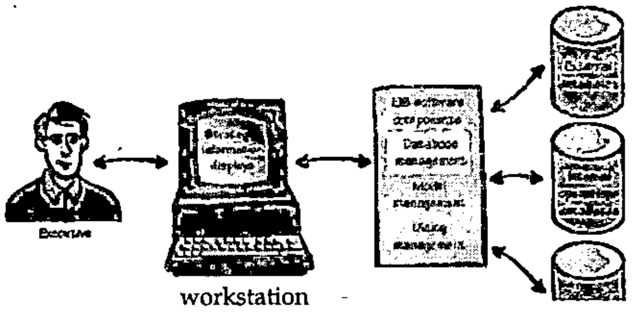


Fig. 9 Illustrating components of an ESS

Custom Software: Organizations develop their own custom software from scratch, rather than use or customize packaged software for three reasons. First, no packaged software meets the required specifications, and modifying existing software is too difficult. Second, the company plans to resell the custom software at a profit. Third, custom software may provide the company with a competitive advantage by providing services for customers, increasing management's knowledge and ability to make good decisions, reducing costs, improving quality, and providing other benefits. Only custom software can provide such a competitive advantage because the competition can easily buy packaged and customized software. Merrill Lynch recognized such a competitive advantage when it planned to invest $1 billion in software (and associated hardware) to allow its financial consultants to operate with mobile computers at customer sites[1].

Custom software is expensive to produce, costly to maintain, subject to bugs, and usually takes many years to develop. Not only does this development time delay the benefit, but it reduces the value of the software as company needs and the competitive environment change constantly. Finally, if software developers can mimic the key features of the software's design and resell it to a company's competitors, they may quickly dilute any competitive advantage the company has gained.

Mutual originally developed CNS because the available packaged software did not meet its needs and because it felt that it could achieve a competitive advantage with CNS. At the time it developed CNS, no software provided an integrated view of a customer across multiple insurance products. CNS gave Mutual agents a competitive advantage by allowing them to cross-sell products where appropriate. Conder has learned that about one-half of all companies in the insurance industry who replace their software use custom software, primarily for these same reasons. By contrast, in the health care industry, less than one-fifth use custom software[2].

Managers who decide to develop custom software must decide whether to use internal company resources or to hire a company that specializes in software development. VARs develop customized software cost-effectively because they have substantial expertise in the base software product, but this advantage disappears for custom software because they cannot use preexisting packages. Small organizations that lack internal resources for software development may create custom software by paying another company to do it. Boston Software Corporation and ECTA Corporation, for example, develop software for companies in the insurance industry[3]. Managers in large organizations usually must use internal IS personnel or let the IS staff decide who will develop the software. Sometimes company policy allows functional managers to seek development bids from both internal and external candidates. Grange Mutual Casualty Company, an Ohio-based insurer with annual premium sales of $500 million, has outsourced some of its development and operations to Policy Management Systems Corporation

The Potential of Various EIS Features

Integrated solutions coordinate the activities of suppliers, distributors, and customers. This integration, called supply-chain management, requires the cooperation of companies outside of the company's own and a secure way of communicating between them. Supply chain management can reduce the costs of a company with $600 million in annual sales by as much as $42 million. Companies using supply chain management needed only two weeks on average to increase production by 20 percent compared to several months without supply-chain management. Also, companies with supply-chain management delivered goods on time 96 percent of the time compared to 83 percent for the average company. Ames, the $200 million maker of garden tools, and Ace Hardware Corporation use Manugistics supply-chain management software to keep track of and replenish Ames's products sold in Ace hardware stores.

The grocery industry calls supply-chain management efficient consumer response (ECR). ECR software breaks barriers between trading partners, such as retailers and manufacturers, and between internal functions such as category management and product replenishment.

Companies such as Land O' Lakes Inc. and Kellogg Company have begun to implement integrated ECR packages to improve internal processes, gain greater efficiency in distribution, and globalize their operations. Oracle and SAP have moved into the ECR market to respond to the growing demand with software derived from their manufacturing packages.

Enterprise-resource-planning (ERP) software products, such as those developed specifically for each industry by SAP, Oracle, Baan, and PeopleSoft, provide seamless support for the supply-chain, value-chain, and administrative processes of a company. They integrate diverse activities internal and external to the company, support many languages and many currencies, and help companies integrate their operations at multiple sites and business units.

Notes

Most ERP applications are customized products. The major vendors begin with an application template that is pre-customized by industry. Then, the vendors, consultants, or the company purchasing the software further customize it to meet individual company needs. Although the ERP vendors build their software to minimize the amount of customization and to simplify the customization process, most large companies spend anywhere from 100 percent to 500 percent of the cost of the software for its customization.

An integrated ERP solution lets managers analyze their operating units as a whole and make decisions from a global perspective. Integrated ERP software eliminates the need to build bridges between applications to share data. For example, sales managers can respond to a manufacturing or ordering plan that might affect the availability of a product. The Monsanto Company, Dow Chemical, and the DuPont Company, all major chemical manufacturers with global operations, have converted their business systems to ERP software products known as R/2 (mainframe based) and R/3 (client/server based), packages produced by SAP AG. The conversion allows them to support simplified business processes as well as communication and collaboration worldwide.

GATX Capital Corporation, a lessor of commercial aircraft,customized the R/3 version of SAP to fit its business. Now it plans to sell its customized work to its competition.16 ERP software often requires companies to change their processes to accommodate the software. ERP lets them introduce improved, redesigned processes because ERP software incorporates the best processes of companies in the industry.

However, when a company has unique processes that give it a competitive advantage, adoption of ERP software can have negative consequences. ERP and any integrated software may not provide as many features or provide them as well as software specifically designed for a particular application, such as inventory management.

Functional managers and general managers need to choose between the best software they can find in each functional area or an integrated package that provides 80 to 90 percent of the features they need in each area. In making this decision, they need to consider the cost of customizing the software to achieve the additional 10 to 20 percent of functionality.

ERP software has not penetrated the service industries because managers are less likely to use systems that their competitors can purchase. Nevertheless, the economic advantages of customized and packaged systems and the success of ERP in manufacturing industries have accelerated the acceptance of integrated packages for a subset of functions in many service industries.

Springer-Miller Systemshas penetrated the hotel market with software that handles all front- and back-office functions. The software performs and integrates such tasks as maintaining guest histories, booking golf tee times and dinner reservations, preparing correspondence, maintaining reservations, and managing multiple properties in a chain.

Hence, we can conclude that modifying vertical application software to deal properly with dates in the year 2020 and beyond is a major problem worldwide. The Gartner Group, a well-respected industry forecaster, estimates the cost to be between $300 and $600 billion. Other estimates run as high as $1.6 trillion.

Notes

According to Viasoft, a consultant involved in these conversions, a typical Fortune 1000 company will spend between $10 and $15 million, consume more than 14 decades of staff time, and require a peak staffing of 48 programmers. Surveys have indicated that 83 percent of companies plan to spend more than 25 percent of their information systems budgets in 1998 and 1999 just to fix their applications for the year 2020

Companies can take three approaches to fixing the problem:

- Use Automated Tools. These tools examine each line of code, identify those that likely refer to dates, and, with some user help, alter them to deal properly with a four-digit year. Vendors and consultants for Tampa Electric, the Florida utility, estimated that it would cost the company about $1 per line to fix its 4.5 million lines code in their 6,000 mainframe programs. After completing a pilot program with automated tools, Tampa Electric estimates that it can complete the task at a substantially lower (although yet unspecified) cost using no more than seven employees over a 22-month period.
- Replace Existing Software with Year 2020 Compliant ERP Products. Because ERP software affects every process in the company, an ERP installation in a large company may take two or three years.

Office Automation and Expert Systems

The term office automation is a method that is used for office activities and makes it possible to process the data using a computer system. An office automation system can be considered as a tool that majorly includes a word processing application, a spreadsheet application, a presentation application, and a database management system.

With implementation, an office automation system will enable the automation of most of the administrative work in the office and would also concentrate on the more repeatable and routine aspects of individual and group work.

An office automation system is a mechanism that allows data transformation from one system to another on its own without human interference and inaccuracies. These tools may be used to capture, organize, and process the data to achieve day-to-day activities. It is an automated process, explicitly supporting business activities and processes. Office automation is intended to provide elements that make it possible to simplify, develop, and automate the organization of the activities of a company or a group of people.

Office automation systems make it simple for office staff to handle day to day organizational activities like E-mail, word processing, electronic filing, scheduling, calendaring, and other technical support resources. Personal digital assistants (PDAs) were also introduced as the concept of groupware apps, which became important when more people started to bring digital assistants such as PDAs. It is made up of word processing, telecommunications, and data processing, which handles office information, official communication, and reports, as well as the processing of documents.

The most commonly used application areas of office automation are as follows -

- Exchange of information.
- Management of administrative records.

- Handling of results.
- Meeting arrangements, preparation, and control of job schedules.

Office automation features

Office automation functionality could include-

- It eliminates the manual effort to complete basic chores.
- Avoiding mistakes by computers or devices.
- Decreasing the time taken to process an object.
- Provides key insights into the process efficiency metrics.
- Gaining greater access to the method and finding possible bottlenecks.
- Controlling the company by making sound decisions based on results.
- Enhancement in business activities with sound improvement.
- Data organization, storage and its management.

Human errors are not only inefficient in that they have to be corrected and can lead to production delays as well as financial losses. For example, an error occurring at the time of financial transaction is a serious mistake and can be a cause of the biggest loss. Human intervention is minimal when the office automation system allows for the transfer of data, rather than taking an active role in the processing.

TCPs, MISs, DSSs, ESSs-the alphabet soup of information systems discussed so far-are designed for managers of various levels. There exist two types .of information systems that are intended for workers of all levels, including those who are not' managers: office automation systems anq expert systems.

Office automation systems. Of fice automation systems (OASs) combine various technologies to red uce the manual labor required in operating and efficient office environment. Used throughout all levels of an organization, OAS technologies include fax, voice mail, e-mail, scheduling software, word processing, and desktop publishing, among others. (See Figure 10)

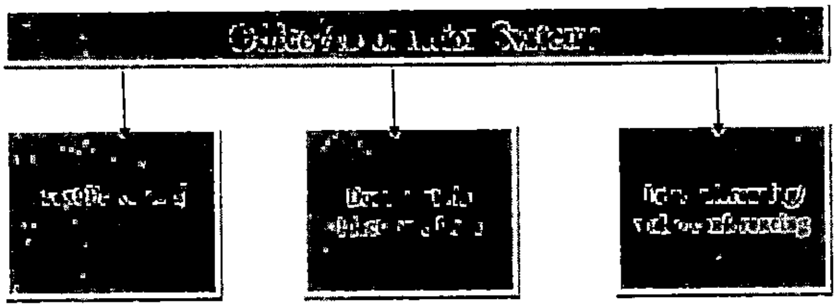


Fig. 10 Office automation systems (The backbone is a network linking these technologies).

The backbone of an OAS is a network—LAN, intranet, extranet-that connects everything. All office functions-dictation, typing, filing, copying, fax, microfilm and records managemeent telephone calls and switchboard operations-are candidates for integration into the network.

Notes

- **Expert systems.** An expert system, or knowledge-based system, is a se.t of interactive computer programs that helps users solve problems that would otherwise require the assistance of a human expert. Expert systems are created on the basis of knowledge collected on specific topics from human specialists, and they imitate the reasoning process of a human being. Remember that expert systems have emerged from the field of artificial int'E1lligence, the branch of computer science that is devoted to the creation of computer systems that simulate human reasoning and sensation.

1.10 SYSTEM DEVELOPMENT LIFE CYCLE

A system is defined as a collection of related components that interact toperform a ta.1k in order to accomplish a goal. A system may not work very well, but it is nevertheless a system. The point of systems analysis and design is to ascertain bow a system works and then take steps to make it better.

An organization's computer-based information system consiSts of hardware, software, people, procedures, and data,. as well as communications set ups. These woi;k togethr to provide people with information for running the organization.

An organization may feel the need for a system due to a variety of reasons. Some examples are:

- A single individual who believes that something badly needs changing is all it takes to get the project rolling.
- An employee may influence a supervisor
- A customer or supplier may get the antion of someone in higher management.
- The management may decide independently to take a look at a system that looks inefficient.
- A steering committee may be formed to decide which of many possible projects should be worked on.

Three types of participants are there in the project as given below:

- **Users.** The system under consideration should always be developed in consultation with users, whether floor sweepers, research scientists, or customers. Indeed, if user involvement in analysis and design is inadequate; the system may fail for lack of acceptance.
- **Management.** Managers within the organization should also be consulted about the system.
- **Technical staff.** Members of the company's information systems (IS) department, consisting of systems analysts and programmers., need to be involved. For one thing, they may have to execute the project. Even if they do not, they will have to work with outside is people contracted to do the job .

Complex projects will require one or several systems analysts. A systems analyst is an information specialist who performs systems analysis, desip, and implementation. The analyst's job is to study the information and communications needs of an organization and determine what changes are required to deliver better information to people who need it. "Better" information means information that is summarized in the acronym "CART"—complete, accurate, relevant, and timely. The systems analyst achieves this goal through the problem solving method of systems analysis and design.

Development Life Cycle

Systems analysis and design is a six-phase problem-solving procedure 'forexamining an information system and improving it. The six phases make up what is known as the systems development life cycle. The systems development life cycle (SDLCJ is the step-by-step process that many organizations follow during systems analysis and design.

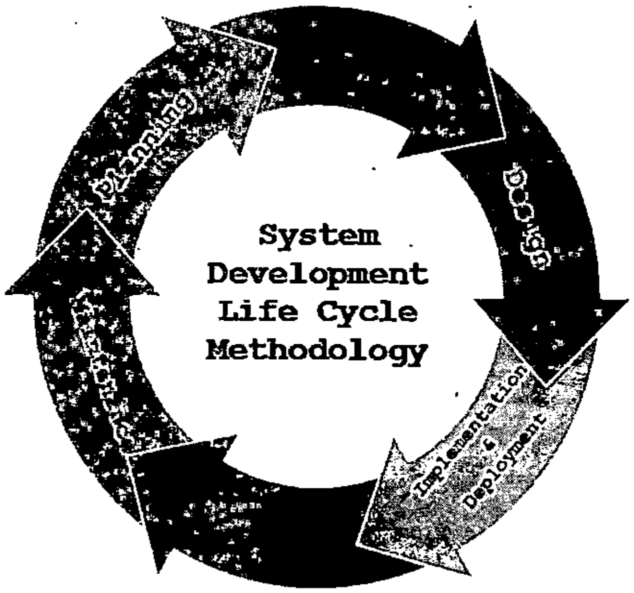

Planning
Design
Implementation & Deployment
System Development Life Cycle Methodology

Fig. 11. The systems development life cycle (SDLC).

Whether applied to a very big company or a three.person engineering business, the six phases in systems analysis and design are as shown in Figure 11. Phases often overlap, and a new one may start before the old one is finished. After the first four phases, management must decide whether to proceed to the next phase. User input and review is a critical part of each phase.

Preliminary Investigation

The objective of Phase I,preliminary investigation, is to cond uct a preliminary analysis, propose alternative solutions,·describe costs and benefits, and submit a preliminary plan with recommendations. These steps are given below:

(i) **Conduct preliminary analysis.** It includes stating the objectives, defining nature and scope of the problem.

Notes

(ii) **Propose alternative solutions:** leave system alone, make it more efficient, or build a new system.

(iii) **Describe costs and benefits of each solution.**

(iv) **Submit a preliminary plan with recommendations.**

Let us explain these in detail:

- **Conduct the preliminary analysis.** In this step, you need to find . out what the organization's objectives are and the nature and scope of the problem under consideration. Even if a problem pertains only to a small segment of the organization, you cannot study it in isola.tion. You need to find out what the objectives of the organization itself are. Then you need to see how the problem being studied fits in with them.
- **Propose alernative solutions.** In delving into the organization's objectives and the specific problem, you may have already discovered some solutions. Other possible solutions can come from interviewing people inside the orgauization, clients or customers affected by it, suppliers and consultants. You can also study what competitors are doing now a days. With this data, you then have three choices. You can leiave the system as is improve it, or develop a new system.
- **Describe the costs and benefits.** Whichever of the three alternatives is chosen, it will have costs and benefits. In this step, you need to indicate what these are. Costs may depend on benefits, which may offer savings. A broad spectrum of benefits may be derived. A process may be speeded up, streamlined through elimination of unnecessary steps, or combined with other processes. Input errors or redundant output may be reduced. Systems and subsystems may be better integrated. Users may be happier with the system. Customers' or suppliers' interactions with the system may be more satisfactory. Security may be improved. Costs may be cut
- **Submit a preliminary plan.** Now you need to wrap up all your findings in a written report. The readers of this report will be the executives who are in a position to decide in which direction to proceed- make no changes, change a little, or change a lot-and how much money to allow the project. You should describe the potential solutions, ;costs, and benefits and mention your recommendations

1.11 SYSTEM ANALYSIS

The objective of Phase 2, system analysis, is to gather data, analyze the data, and write a report. In this second phase of the SDLC, you will follow the course that management has indicated after having read your Phase 1 feasibility report.

We are assuming that they have ordered you to perform Phase 2-to do a careful analysis or study of the existing. system in order to understand how the new system you proposed would differ. This analysis will also consider how people's positions and tasks will have to change if the new system is put into effect. The steps are given below:

(i) Gather data, using tools of written documents, interviews, questionnaires, and observations.

(ii) Analyze the data, using modelling tools: grid . charts, decision tables, data flow diagrams, systems flow charts, connectivity diagrams.

(iii) Write a report.

Let us explain these in detail:

- **Gather data.** In gathering data, you will review written documents, interview employees and managers, develop questionnaires, and observe people and processes at their place of work.
- **Analyze the data.** Once the data has been gathered, you need to come to grips with it and analyze it. Many analytical tools, or modelling tools, are available. Modelling tools enable a systems analyst to present graphic, or pictorial, representations of a system. An example of a modelling tool is a data flow diagram (DFD), which graphically shows the flow of data through a system-that is, the essentiai processes of a system, along with inputs, outputs and files. (See Figure 12).

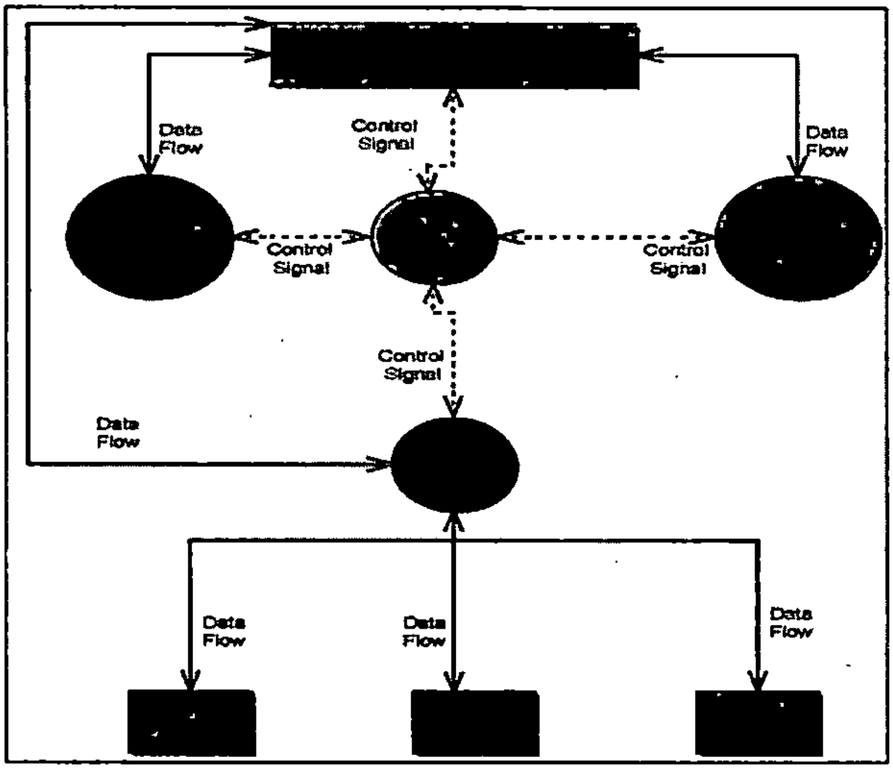


Fig. 12. Data flow diagram.

- **Write a report.** After completion of the analysis, you need to document this phase. This report to management should have three parts:
  - It should explain how the existing system works.
  - It should explain the problems with the existing system.
  - It should describe the requirements for the new system and make recommendations on what to do next.

Notes

At this stage, not a lot of money will have been spent on the systems analysis and design project. Ifthe costs of going forward appear prohibitive, this is a good time for the managers reading the report to call a halt. Otherwise, you will be asked to go ahead to Phase 3.

1.12 SYSTEM DESIGN

The objective of Phase 3, sytems design is to do a preliminary design and then a detail design, and write a report. The steps are given below:

(i) Do apreliminary design, usirig CASE (Computer-Aided Soware Engineering) tools, prototyping tools, and project management software, among others.

(ii) Do a detail design, defining requirements for output, input, storage, and processing and system controls and backup.

(iii) Write a report.

In this third phase of the SDLC, you will essentially create a "rough draft" and then a detail draft" of the proposed information system.

Let us explain the above mentioned steps in detail:

- **Do a preliminary design.** A preliminary design describes the general functional capabilities of a proposed information system. It reviews the system requirements and then considers major components of the system under consideration. Usually several alternative systems (called candidates) are considered, and the costs and the benefits of each are evaluated.

  Some tools that may be used in the design are CASE tools and project management software.

  CASE (Computer-Aided Software Engineering) tools are programs that automate various activities of the SDLC in several phases. This technology is intended to speed up the process of developing systems and to improve the quality of the resulting systems. These tools, which are also known as automated design tools, may be used at other stages of the SDLC as well. Some examples of such programs are Excelerator, Iconix, System Architect, and Powerbuilder.

  Prototyping refers to using workstations, CASE tools, and other software applications to build t:Jorking models of system components, so that they can be quickly tested and evaluated. Thus, a prototype is a limited working system developed to test out design concepts. A prototype, which may be constructed in just a few days, allows users to find out immediately how a change in the system might benefit them. For example, a systems analyst might develop a menu as a possible screen display, which users could try out. The menu can then be redesigned or fine-tuned, if necessary.

  Project management software consists of programs used to plan, schedule, and control the people, costs, and resources required to complete a project on time.

- *Do a detail design:* A detail design describes how a proposed information system will deliver the general capabilities described in the preliminary design. The detail design, usually considers the following parts of the system in this order output requirements input requirements storage requirements processing requirements, and system control and back up.

- *Write a report.* All the work of the preliminary and detail designg will end up in a large, detailed report. When you hand over this report to senior management you will probably also make some sort of presentation or speech.

1.13 SYSTEM DEVELOPMENT

Once the system design phase is over, next system development is begun. The necessary hardware and software are acquired or developed and the system is tested. The fourth-phase of the six-phase SDLC is made up of many activities.

This phase is expensive because so many peoples are involved in the process; it is time-consuming because of a lot of work has to be completed. Regardless of the methodology used, once coding and testing are complete the system is ready to "go live".

If the hardware and software are available commercially, they are purchased of leased. Although it is much cheaper to purchase commercially available software (called off-the-shelf software), these programs do not always fit the needs of the organization developing the system.

If not, custom software must be developec or a commercially available program must be modified by programmers in-house If the work is performed by outside contractors it is called outsourcing

The major activities we are concerned with in this unit are coding, acquiring hardware, and testing. The purpose of these steps is to convert the physical system specifications into working and reliable software and hardware. Coding and testing may have already been completed by this point if agile methodology have been followed. Using a plan-driven methodology, coding and testing are often done by other project team members besides analysts, although analysts, may do some programming. In any case, analysts are responsible for ensuring that all of these activities are properly planned and executed.

The Processes of Coding, Acquiring Hardware, and Testing

Coding is the process whereby the physical design specifications created by the analysis team are turned into working computer code by the programming team. Depending on the size and complexity of the system, coding can be an involved, intensive activity. Regardless of the development methodology followed, once coding has begun, the testing process can begin and proceed in parallel.

As each program module is produced, it can be tested individually, then as part of larger program, and then as part of a larger system. You will learn about the different strategies for testing later in the unit. We should emphasize that although testing is done during development, an analyst must begin planning for testing earlier in the project. Preliminary investigation involves determining what needs to be tested and collecting test data. This is often done during the analysis phase because testing requirements are related to system requirements.

Outcomes from Coding, Acquiring Hardware, and Testing

Table 1 shows the deliverables from the coding, acquiring hardware, and testing processes. Some object-oriented languages, such as Eiffel, provide for documentation to be extracted automatically from software developed in Eiffel. Other languages, such as Java, employ specially designed utilities, such as JavaDocs, to generate documentation from the source code.

Other languages will require more effort on the part of the coder to establish good documentation. But even well-documented code can be mysterious to maintenance programmers who must maintain the system for years after the original system was written and the original programmers have moved on to other jobs. Therefore, clear, complete documentation for all individual modules and programs is crucial to the system's continued smooth operation.

Table 1. Deliverables for Coding, Acquiring Hardware, and Testing

1. Coding
(a) Code
(b) Program documentation
2. Acquiring Hardware
(a) Acquire' or Upgrade.
3. Testing
(a) Test scenarios (test plan) and test data
(b) Results of program and system testing

Increasingly, CASE tools are used to maintain the documentation needed by systems professionals. The results of program and system testing are important deliverables from the testing process because they document the tests as well as the test results. For example,

- What type of test was conducted?
- What test data were used?
- How did the system handle the test?

The answers to these questions can provide important information for system maintenance because changes will require retesting and similar testing procedures will be used during the maintenance process.

An analyst's job is to ensure that all of these deliverables are produced and are done well. An analyst may produce some of the deliverables, such as test data, user guides, and an installation plan; for other deliverables, such as code, he/she may only supervise or simply monitor their production or accomplishment.

The extent of his/her implementation responsibilities will vary according to the size and standards of the organization he/she works for, but his/her ultimate role includes ensuring that all the implementation work leads to a system that meets the specifications developed in earlier project phases.

Where Programming Fits in the SDLC

Every type of software (pre-written software, or a customized software, or a public domain software) has to be developed by someone before we can use it. Software or program must be understood properly before its development and use. A program is a list of instructions that the computer must follow in order to process data into information. The instructions consist of statements used in a programming language; such as C or C++.

Examples are programs that do word processing, desktop publishing, or railway reservation. The decision whether to buy or develop a program forms part of Phase 4 in the systems development life cycle.

Figure 13 illustrates this once the decision is made to develop a new system, the programmer starts his/her work. The Phase 4 of the six-phase SDLC includes a five step procedure of its own as shown in the bottom. These five steps constitute the problem solving or software development process known as programming. Programming also known as software engineering, is a multistep process for creating that list of instruct.ions (i.e, a program fot the computer).

The five steps are given below :

1. Clarify the problem-include needed output-input, processing requirements.
2. Design a solution-use modelling tools to chart the program.
3. Code the program-use a programming language's syntax, or rules, to write the program.
4. Test the program, get rid of any logic errors, or "bugs", in the program ("debug" it).
5. Document and maintain the program-include written instructions for users, explanation of the program, and operating instructions.

Coding sitting at the keyboard and typing words into a computer is what many people imagine programming to be. As we see, however, it is only one of the five steps. Coding consists of translating the logic requirements into a programming language—the letters, numbers and symbols that make up the program.

Clarify the Programming Needs

The problem clarification step consists of six sub-steps-clarifying program objectives and users, outputs, inputs, and processing tasks; studying the feasibility of the program; and documenting the analysis. Let us consider these six substeps.

(i) **Clarify Objectives and Users.** We solve problems all the time. A problem might be deciding whether to take a required science courae this term or next, or selecting classes that allow us also to fit a job into our schedule. In such cases, we are specifying our objectives. Programming works the same way. We need to write a statement of the objectives we are trying to accomplish—the problem we are trying to solve. If the problem is that our company's systems analysts have designed a new computer-based payroll processing program and brought it to us as the programmer, we need to clarify the programming needs.

Notes

We also need to make sure us know who the users of the program will be. Will they be people inside the company, outside, or both ? What kind of skills will they bring?

(ii) **Clarify Desired Outputs.** Make sure to understand the outputs-what the system designers want to get out of the system-before we specify the inputs. For example, what kind of hardcopy is wanted? What information should the outputs include ? This step may require several meetings with systems designers and users to make sure we are creating what they want.

(iii) **Clarify Desired Inputs.** Once we know the kind of outputs required, we can then think about input. What kind of input data is needed? What form should it appear in? What is its source?

(iv) **Clarify the Desired Processing.** Here we make sure to understand the processing tasks that must occur in order for input data to be processed into output data.

(v) **Double-Check the Feasibility of Implementing the Program.** Is the kind of program we are supposed to create feasible within the present budget ? Will it require hiring a lot more staff ? Will it take too long to accomplish?

Sometimes programmers decide they can buy an existing program and modify it rather than write it from scratch.

(vi) **Document the Analysis.** Throughout program clarification, programmers must document everything they do. This includes writing objective specifications of the entire process being described.

Design the Program

Assuming the decision is to make, or custom-write, the program, we then move on to design the solution specified by the systems analysts. In the program design step, the software is designed in three mini-steps. First, the program logic is determined through a top-down approach and modularization, using a hierarchy chart. Then it is designed in detail, either in narrative form, using preudocode, or graphically, using flowcharts.

Today most programmers use a design approach called structured programming. Structured programming takes a top-down approach that breaks programs into modular forms. It also uses standard logic tools called control structures.

Code the Program

Once the design has been developed, the actual writing of the program begins. Writing the program is called coding. Coding is what many people think of when they think of programming, although it is only one of the five steps. Coding consists of translating the logic requirements from pseudocode or flowcharts into a programming language – the letters, numbers, and symbols that make up the program..

(i) *Select the Appropriate Programming Language.* A programming language is a set of rules that tells the computer what operations to do. Examples of well-known programming languages are C, C++, COBOL, visual Basic and JAVA. These are called high-level languages. Not all languages are appropriate for all uses. Some, for example, have strengths in mathematical and statistical processing. Others are more appropriate for database management. Thus, in choosing the language, we need to con.sider what purpose the program is designed to serve and what languages are already being used in our organization or in our field.

(ii) **Follow the Syntax.** In order for a program to work, we have to follow the synta. the rules of the programming language. Programming languages have their own gram. mar just as human languages do. But computers are probably a lot less forgiving if we use these rules incorrectly.

Test the Program

Program testing involves running various tests and then running real-world data to make sure the program works. Two principal activities are desk-checking and debugging. These steps are known as alpha-testing.

(i) **Perform Desk-Checking.** Desk -check ing is simply reading through, or checking, the program to make sure that it's free of errors and that the logic works. In other words, desk-checking is like proofreading. This step could be taken before the program is actually run on a computer.

(ii) **Debug the Program.** Once the program has been desk-checked, further errors, or "bugs", will doubtless surface. To debug means to detect, locate, and remove all errors in a computer program. Mistakes may be syntax errors or logical errors. Syntax errors are caused by typographical errors and incorrect use of the programming language. Logic errors are caused by incorrect use of control structures. Programs called diagnostics exist to check program syntax and display syntaicerror messages. Diagnostic programs thus help identify and solve problems.

(iii) **Run Real-World Data.** After desk-checking and debugging, the program may run fine-in the laboratory. However, it needs to be tested with real data; this is called beta testing. Indeed, it is even advisable to test the program with bad data-data that is faulty, incomplete, or in overwhehning quantities—to see if you can make the system crash. Many users, after all, may be far more heavy-handed, ignorant and careless than programmers have anticipated.

Several trials using different test data may be required before the programming team is satisfied that the program cart be released. Even then, some bugs a may persist, because there comes a point where the pursuit of errors is uneconomical. This is one reason why many users are nervous about using the first version (version 1.0) of a commercial software package.

Document and Maintain the Program

Writing the program documentation is the fifth step in programming. The resulting documentation consists of written descriptions of what a program is and how to use it. Documentation is not just an end-stage process of programming. It has been (or should have

Notes

been) going on throughout all programming steps. Documentation is needed for people who will be using or be involved with the program in the future.

Documentation should be prepared for several different kinds of readers-users, operators and programmers.

(i) **Prepare User Documentation.** When we buy a commercial software package, such as a spreadsheet, we normally get a manualwith it. This is user documentation.

(ii) **Prepare Operator Documentation.** The people who run large computers are called computer operators. Because they are not always programmer, they need to be told what to do when the program malfunctions. The operator documentation gives them this information.

(iii) **Write Programmer Documentation.** Long after the original programming team has disbanded, the program may still be in use. If, as in often the case, a fourth of the programming staff leaves every year, after 4 years there could be a whole new bunch of programmers who know nothing about the software. Program documentation helps train these newcomers and enables them to maintain the existing system.

(iv) **Maintain the Program.** Maintenance includes any activity designed to keep programs in working condition, error-free, and up to data-adjustments, replacements, repairs, measurements, tests and so on. The rapid changes in modern organizations in products, marketing strategies accounting systems, and so on-are bound to be reflected in their computer systems. Thus, maintenance is an important matter, and documentation must be available to help programmers make adjustments in existing systems.

The five steps of the programming process are summarized in **Table 2.**

Table 2. Summary of the Five Programming Steps

Step	Activities
Step 1: Problem definition	1. Specify program objectives and program users. 2. Specify output requirements. 3. Specify input requirements. 4. Specifyprocessing requirements. 5. Study feasibility of implementing program: 6. Document the analysis.

Step 2: Program design	1. Determine program logic through top-dow approach and modularization, using a hierarchy chart. 2. Design details using pseudocode and/or using flowcharts, preferably on the basis of control structures. 3. Test design with structured walk through.
Step 3: Program coding	1. Select the appropriate high-level programming language. 2. Code the program in that language, following the syntax carefully.
Step 4: Program testing	1. Desk-check the program to discover errors. 2. Run the program and debug it (alpba testing). 3. Run real-world data (beta testing.).
Step 5: Program documentation and maintenance	1. Prepare user documentation. 2. Write operator documentation. 3, Write programmer documentation. 4. Maintain the program

In Phase 4, systems develop ment, the systems analyst or others in the organization, develop or acquire the software, acquire the hard ware, and then test the system. The steps are given below:

(i) Acquire software.

(ii) Acquire hard ware.

(iii) Test the system.

Depending on the size of the project, this phase will probably involve the organization in spending substantial sums of money. It could also involve spending a lot of time. However, at the end you should have a workable system.

Let us explain the above mentioned steps in detail:

- Develop or acquire the software. During the design stage the systems analyst may have had to address what is called the "make-or-buy" decision, but that decision certainly cannot be avoided at this stage. In the make or-buy decision, you decide whether you have to create a program have it custom- written-or buy it, meaning simply purchase an existing software package. Sometimes programmers decide they can buy an existing program and modify it rather than write it from scratch.

- If you decide to create a new program, then the question is whether to use. the organization's own staff programmers or to hire outside contract programmers (outsource it). Whichever way you go, the task could take many months.
- **Acquire hardware.** Once the software has been chosen, the hardware to run it must be acquired or upgraded. It's possible your new system will not require any new hardware. It's also possible that the new hardware will cost a huge amount and involve many items: microcomputP.rs, mainframes, monitors, modems, and many other devices. The organization may find it's better to lease rather than to buy some equipment, especially since, as we know (Moore's law), chip capability has traditionally doubled every 18 months.
- **Test the system.** With the software and harciware acquired, you can now start testing tne system. Testing is usually done in two stages: unit testing, then system testing.
- In unit testing, the performance of individual parts is examined, using test (made up, or sample) data. If the program is written as a collaborative effort by multiple programmers, each part of the program is tested separately.
- In system testing, the parts are linked together, and test riata is used to see if the parts work together. At this point, actual organization data may be used to test the system. The system is also tested with erroneous and massive amounts of data to see if the system can be made to fail ("crash").

At the end of this long process, the organization will have a workable information system, one ready for the implementation phase.

1.14 SYSTEMS IMPLEMENTATION

Systems implementation is the process of: defining how the information system should be built (i.e., physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e., quality assurance).

Whether the new information system involves a few handheld computers, an elaborate telecommunications network, or expensive mainframes, the fifth phase will involve: some close coordination in order to makethe system not just workable but successful. Phase 5, systems implementation, consists of converting the hard ware, soft ware, and files to the new system and traininq the users. The steps are giyen below:

(i) Convert hardware, so'ftware, and files through one of four types of conversions: direct, parallel, phased, or pilot:

(ii) Compile final documentation.

(iii) Train the users.

Let us explain the above mentioned steps in detail:

- **Convert to the new system.** Conversion, the process of transition from an old information system to a new one, requires converting hardware, software, and files. There are four strategies for handling conversion: direct, parallel, phased, and pilot.

- **Direct Implementation** means that the user simply stops using the old system and starts using the new one. The risk of this method should be evident: What if the new system doesn't work? If the old system has truly been discontinued, there is nothing to fall back on.
- **Parallel implementation** means that the old and new systems are operated side by side until the new system has shown it is reliable, at which time the old system is discontinued. Obviously there are benefits in taking this cautious approach. If the new system fails, the organization can switch back to the old one. The difficulty with this method is the expense of paying for the equipment and people to keep two systems working at the same time.

Phased implementation means that parts of the new system are phased in separately-either at different times (parallel) or all at once in groups (direct).

Pilot implementation means that the entire system is tried out but .only by some users. Once the reliability has been proved, the system is implemented with the rest of the intended users. The pilot approach still has its risks, since all of the users of a particular group are taken off the old system. However, the risks are confined to a small part of the organization.

Compile final documentation. Documentation of a system consists of written description of system's specification, its design, code, operating procedures etc. It is quite useful to users and maintenanGe programmers. Documentation can be broadly classified into two types:

Documentation for users—it describes how to use the system.

Documentation for Maintenance Programmers—it is also called technical documentation and used for system modification at some later stages. Documentation of a system must start with the system definition. It is very difficult to think about the documentation in the end.

Train the users. Various tools are available to familiarize users with a new system-from documentation (instruction manuals) to videotapes to live class.es to one-on-one, side-by-side teacher-student training. Sometimea training is done by the organization's own staffers, at other times it is contracted out.

The Process of Installation, Documenting the System

Training Users, and, Supporting Users

Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, documentation, and work procedures to those consistent with the new system. Users must give up the old ways of doing their jobs, whether manual or automated, and adjust to accomplishing the same tasks with the new system. Users will sometimes resist these changes, and an analyst must help them adjust. However, he/she cannot control all the dynamics of user-system interaction involved in the installation process.

Although the process of documentation proceeds throughout the life cycle, it recei. ves formal attention during the implementation phase because the end of implementation

Notes

largely marks the end of the analysis team's involvement in systems development. As the team is getting ready to move on to new projects, all the analysts working on the system need to prepare documents that reveal all of the important information they have learned about this system during its development and implementation, There are two audiences for this final documentation :

1. the information systems personnel who will maintain the system throughout its productive life and
2. the people who will use the system as part of their daily lives.

The analysis team in a large organization can get help in preparing documentation from specialized staff in the information systems department.

Larger organizations also tend to provide training and support to computer users throughout the organization. Some of training and support is very specific to particular application systems, whereas the rest is general to particular operating systems or off-the-shelf software packages. For example, it is common to find courses on Microsoft Windows and WordPerfect in organizationwide training facilities.

Analysts are mostly uninvolved with general training and support, but they do work with corporate trainers to provide training and support tailored to particular computer applications they have helped to develop. Centralized information system training facilities tend to. have specialized staff who can help with training and support issues. In smaller organizations that cannot afford to have well-staffed centralized training and support facilities, fellow users are the best source of training and support users have, whether the software is customized or off the shelf.

Installation

The process of moving from the current information system to the new one is called installation. All employees who use a system, whether they were consulted during the development process or not, must give up their reliance on the current system and begin to rely on the new system. Four different approaches to installation have. emerged over the years direct, parallel, single location, and phased.

The approach an organization decides to use will depend on the scope and complexity of the change associated with the new system and the organization's risk aversion (strong dislike).

Direct Installation

The direct, or abrupt, approach to installation (also called "cold turkey") is as sudden as the name indicates: The old system is turned off and the new system is turned on. Under direct installation, system users are at the mercy of the new system. Any errors resulting from the new system will have a direct. impact on the users and how they do their jobs and, in some cases depending on the centrality of the system to the organization.on how the organization performs its business. If the new system fails, considerable delay may occur until the old system can again be made operational and business transactions are reentered to make the database up-to-date.

For these reasons, direct installation can be very risky. Further, direct installation requires a complete installation of the whole system. For a large system, this may mean a long time until the new system can be installed, thus delaying system benefits or even missing the opportunities that motivated the system request.

On the other hand, it is the least expensive installation method, and it creates considerable interest in making the installation a success. Sometimes, a direct installation is the only possible strategy if there is no way for the current and new systems to coexist, which they must do in some way in each of the other installation approaches.

Parallel Installation

Parallel installation is as riskless as direct installation is risky. Under parallel installation, the old system continues to run alongside the new system until users and management are satisfied that the new system is effectively performing its duties and the old system can be turned off. All of the work done by the old system is concurrently performed by the new system. Outputs are compared (to the greatest extent possible) to help determine whether the new system is performing as well as the old.

Errors discovered in the new system do not cost the organization much, if anything, because errors can be isolated and the business can be supported with the old system. Because all work is essentially done twice, a parallel installation can be very expensive; running two systems implies employing (and paying) two staffs to not only operate both systems, but also to maintain them. A parallel approach can also be ronfusing to user 3 because they must deal with both systems. As with direct installation, there can be a considerable delay until the new system is completely ready for inf'tallation.

A parallel approach may not be feasible, especially if the users of the system (such as customers) cannot tolerate redundant effort or if the size of the system (number of users or extent of features) is large.

Single-Location Installation

Single-location installation, also called location or pilot installation, is a middle of-the-road approach compared with direct and parallel installation. Rather than convert all of the system in o.rganization at once, single-location installation involves changing from the current to the new system in only one place or in a series of separate sites over time illustrates this approach for a simple situation of two locations).

The single location may be a branch office, a single factory, or one department, and the actual approach used for installation in that location may be any of the other approaches. The key advantage to. single location installation is that it limits potential damage and potential cost by limiting the effects to a single site.

Once management has determined that installation has been successful at one location, the new system may be deployedJn the rest of the organization, possibly continuing with installation at one location at a time. Success at the pilot site can be used to convince reluctant personnel at other sites that the system can be worthwhile for them as well. Problems with the system (the actual software as well as documentation, training, and support) can be resolved before deployment to other sites of the organization. Even though the single-location approach may be simpler for users, it still places a large burden on is (Information Systems) staff to support two versions of the system.

On the other hand, because problems are isolated at one site at a time, is staff can devote all of their efforts to the success at the pilot site. Also, if different locations require sharing of data, extra programs will need to be written to synchronize the current and new systems; although this will happen transparently to users, it is extra work for is staff. As with each of the other approaches (except phased instailation), the whole system is installed; however, some parts of the organization will not gel the benefits of the new system until the pilot installation has been completely tested.

Phased Installation

Phased installation, also known as staged installation, is an incremental approach. With phased installation, the Iiew system is brought online in functional components; different parts of the old and new systems are used in cooperation until the whole new system is installed. shows the phase-in of the first two modules of a new system.) Phased installation, like single-location installation, is an attempt to limit the organization's exposure to risk, whether in terms of cost or disruption of the business. By converting gradually, the organization's risk is spread out over time and place.

Also, a phased installation allows for some benefits from the new system before the whole system is ready. For example, new data-capture methods can be used before all reporting modules are ready. For a phased installation, the new and replaced systems must be able to coexist and probably share data. Thus, bridge programs connecting old and new databases and programs often must be built.

Sometimes, the new and old systems are so incompatible (built using totally different structures) that pieces of the old system cannot be incrementally replaced, so this strategy is not feasible. A phased installation is akin to bringing out a sequence of releases of the system. Thus, a phased approach requires careful version control, repeated conversions at each phase, and a long period of change, which may be frustrating and confusing to users. On the other hand, each phase of change is smaller and more manageable for all involved

Planning Installation

Each installation strategy involves converting not only software, but also data and (potentially) hardware; documentation, work methods, job descriptions, offices and other facilities, training materials, business forms, and other aspects of the system.

For example, it is necessary to recall or replace all the current system documentation and business forms, which suggests that the is department must keep track of who has these items so that they can be notified and receive replacement items. In practice, an analyst will rarely choose a single strategy to the exclusion of all others most installations will rely on a combination of two or more approaches. For example, if he/she selects a single-location strategy, he/she has to decide how installation will proceed there and at subsequent sites. Will it be direct, parallel, or phased?

Of special interest in the installation process is the conversion of data. Because existing systems usually contain data required by the new system, current data must be made error

-free, unloaded from current files, combined with new data, and loaded into new files. Data may need to be reformatted to be consistent with more advanced data types supported by newer technology used to build the new system.

New data fields may have to be entered in large quantities so that every record copied from the current system has all the new fields populated. Manual tasks, such as taking a physical inventory, may need to be done in order to validate data before they are transferred to the new files. The total data conversion process can be tedious.

Furthermore, this process may require that current systems be shut off while the data are extracted so that updates to old data, which would contaminate the extract process, cannot occur.

Any decision that requires the current system to be shut down, in whole or in part, before the replacement system is in place must be done with care. Typically, off hours are used for installations tha(require a lapse in system support. Whether a lapse in service is required or not, the installation schedule should be announced to users well in advance to let them plan their work schedules around outages in service and periods when their s.ystem support might be erratic.

Successful installation steps should also be announced, and special procedures put in place so that users can easily inform the. analyst of problems they encounter during installation periods. An analyst should also plan for emergency staff to be available in case of system failure so that business operations can be recovered and made operational as quickly as possible.

Another consideration is the business cycle of the organization. Most organizations face heavy workloads at particular times of year and relatively light loads at other times. A well-known example is the retail industry, where the busiest time of year is the fall, right before the year's major gift-giving holidays. An analyst wouldn't want to schedule installation of a new point-of-sale system to begin December 1for a department store. Make sure he/she understands the cyclical nature of the business he/she is working with before he/she schedules installation.

Planning for installation may begin as early as the analysis of the organization supported by the system. Some installation activities, such as buying new hardware, remodeling facilities, validating data to be transferred to the new system, and collecting new data to be loaded into the new system, must be done before the software installation can occur. Other the project team leader is responsible for anticipating all installation tasks and assigns responsibility for each to different analysts.

Each installation process involves getting workers to change the way they work. As such, installation should be looked at not as simply installing a new computer system, but as an organizational change process.

1.15 POST IMPLEMENTATION REVIEW AND MAINTENANCE

In this section, we discuss systems maintenance, the largest systems development expenditure for many organizations. In fact, more programmers today work on maintenance activities than work on new development. Your first job after graduation/post graduation may very well be as a maintenance programmer/analyst. This disproportionate distribution of maintenance

Notes

programmers is interesting because software does not wear out in a physical way as do buildings and machines.

There is no single reason why software is maintained; however, most reasons relate to a desire to evolve system functionality in order to overcome internal processing errors or to better support changing business requirements. Thus, maintenance is a fact of life for most systems. It means that maintenance can begin soon after the system is installed.

As with the initial design of a system, maintenance activities are not limited only to software changes, but include changes to hardware and business procedures. A question many persons have about maintenance relates to how long organizations should maintain a system. Five years? Ten years? Longer?

This is not an easy question to answer, but it is most often an issue of economics. In other words, at what point of time does it make financial sense to discontinue evolving an older system and build or purchase a new one? The focus of a great deal of upper is (Information Systems) management attention is devoted to assessing the trade-offs between maintenance and new development. In this unit, we will provide you with a better understanding of the maintenance process and describe the types of issues that must be considered. when maintaining systems.

In this unit, we also briefly describe the systems maintenance process and the deliverables and outcomes from this process. This is followed by a detailed discussion contrasting the types of maintenance, an overview of critical management issues, and a description of the role of CASE and automated development tools in the maintenance'rocess. Finally you will learn about review of system performance (syst!i!m audit) and system audit report.

Maintaining Information Systems

Once an mformation system is stalled, the system is essentially in the maintenance phase of the.systems development life cycle. When a system is in the maintenance phase, some person within the systems development group is responsible fot collecting maintenance requests from system users and other interested parties such as system auditors, data center and network management staff, and data analysts.

After collecting the requests, each request is analyzed to better understand how it will alter the system and what business benefits and necessities will result from such a change. If the change request is approved, a system change is designed and then implemented. As with the initial development of the system, implemented changes are formally reviewed and tested before installation into operational systems.

The Process of Maintaining Information Systems

Figure 14, illustrates that the maintenance phase is the last phase of the systems development life cycle. It is here that the SDLC becomes a cycle, with the last activity leading back to the first. This means that the process of maintaining an information system is the process of returning to the beginning of the SDLC and repeating development steps until the ch!inge is implemented.

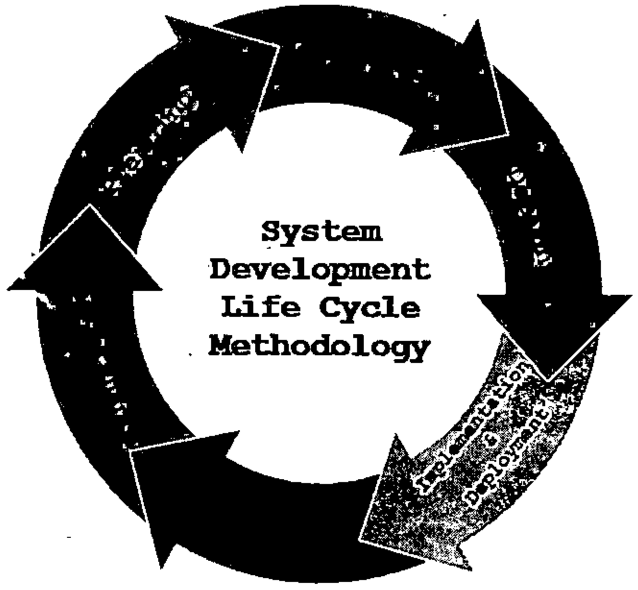

System Development Life Cycle Methodology
Implementation & Deployment

Fig. 14. System Development Life Cycle (SDLC)

Phase 6, systems maintenance, adjusts and improves the system by having system audits and periodic evaluations and by making changes based on new conditions.

The sixth phase is to keep the system running through system audits and periodic evaluations. Even with the conversion accomplished and the users trained, the system would not just run itself. There is a sixth-and never-ending-phase in which the information system must be monitored to ensure that it is successful.

Maintenance includes not only keeping the machinery running but also updating and upgrading the system to keep pace with new products, services, customers, government regulations, and other requirements. So we can conclude that "the better the system maintenance, the best the user satisfaction.

1.16 SYSTEM DOCUMENTATION CONSIDERATIONS

Principles of System Documentation

An often overlooked and underrated part of a system is its documentation. Documentation is an integral part of the various phases of the life cycle and is produced as part of the phase. Unfortunately, many system persons see documentation as a formality to be performed at the end of the stage rather than work to be done as an integral part of a stage. This results in ineffective, poorly written and incomplete documentation, which is not usable.

Documentation is a major means of communication. It is the means of communication, which survives over time after the analyst has left the project and the team has disbanded-the documentation is what remains. It is through the documents that users learn about the system

Notes

and they need to refer to the documents to use the system. Documentation forms a written record for the work; it establishes design and performance criteria for the project phases.

Care needs to be taken to create good and effective documentation.

Documentation should be created as part of the process and not written after the fact. Documentation is the product of all phases. It should be a working by-product throughcsut the life cycle. The tendency to "postdocument" (document after the fact) should be avoided. All documentation should follow certain standards which are prescribed out for the project or the organization.

Documentation should be reviewed and approved for release. It should be available to all authorized persons who may need to refer to it. Obsolete documents should not be in circulation to avoid confusion.

The procedure to be used to request for change in documentation, to evaluate and process such changes, to review changes made and to release the modified documents should be clearly laid. Unauthorized persons should not be able to change documentation. All changes made to the documents and the reasons why they were made should be noted as part of the documents Standard used to create documents typically address the following :

- Title page and documents id, along with other identifiers like project name
- Versions control information
- Names of author, reviewer, approver
- Date of release
- Version number
- Change control history
- Table of contents, figures and tables
- Scope of the documents
- Expected reader profile
- Definitions and acronyms
- Other documents to refer to (specific references)
- Overview/introduction
- Summary/conclusions as executive summary, if relevant
- Main body of the documnts
- Supporting appendices.

The style used should ensure that each art of the documents can be referred to (has some identifier). The language should be

- Tuned to the reader profile
- Clear
- Concise
- Comprehensive
- Courteous

- Precise
- Simple
- Flowing properly
- Easy to read and understand
- Maintaining continuity

Documents should include all necessary illustrations and tables. References should be provided for the ease of the user. The style should be consistent. The headings, style and general appearance should be consistent. The hierarchy of the documents should be clear to the reader.

Technical editors are sometimes used to ensure that the documents are well written. Typically, organizations have standards for each type of documents, For example, the organization may have a standard for requirements specification which give each of the topics to be addressed and the order in which they are to be put in the documents. These standards are derived using industry standard organizations based on quality models previous experiences and the opinions of senior staff in the organization.

Types of Documentation and their Importance

Documentation is created at each phase of a project. The major types of documentation and importance are given as follows:

Documents that form inputs to the development process but are not created are not covered here.

Enforcing Documentation Discipline in an Organization

As mentioned earlier, unfortunately, documentation is usually done after the fact. In fact, some teams actually have a "documentation phase" after the rest of the work is over, in which the documents are generated. The discipline of documentation has to be enforced in an organization.

To ensure that the culture of documentation is an integral part of the system team's work habit, it is necessary that documentation be considered an integral part of work. This implies that:

- Budgets should include documentation effort
- Procedures are set up to create, review documents.
- Management does not say "as time is running short, just create the system. We will create the document later.

To reinforce the integral aspect of documentation, "completion" of an activity should include completion of the associated document. For example,

- A phase should not be considered complete unless the documentation is complete
- A progrshould not be considere complete unless it has required comment lines.
- A unit test activity should not be considered complete unless the test packs and records are correctly available.

Notes

Also, the next phase should not be starte unless a phase is complete including the documentation. Documentation completion (of the required quality should be, one of the aspects considered while evaluating work.

To make documentati happen, the task of documenting should be made easy. To enable persons document easily and pro organization should have in place standards which let the staff know how the document should be written. This ensures the good quality documents are created easily.

Training in technical writing and workshops help system persons write better. A technical editor may also be used to provide a more professional look.

To should be available to simplify the task. Proper word processors and drawing tools, with appropriately set up templates help in making the task easier and reducing the resistance.

Different Approaches to Improving Development

In the continuing effort to improve the systems analysis and design process, several different approaches have been developed. We will describe some important approacbea here. Attempts to make systems development less of an art and more of a science are usually referred to as systems engineering or software engineering. As the names indicate, rigorous engineering techniques have been applied to systems development.

Although the application of some engineering processes to software development, such as the strict waterfall SDLC approach, have been criticized, one very influential practice success. fully borrowed from engineering is called prototyping... . We will discuss prototyping first, and then CASE tools.

Prototyping

Designing and building a scaled-down but functional version of a desired system is known a.s prototyping. A prototype can be built with any computer language or development tool, but special prototyping tools have been developed to simplify the process. A prototype can be developed with visual development tools; with the query, screen, and report design tools of a database management system; and with CASE tools.

Using prototyping as a development technique (See Figure 15); the analyst works with users to determine the initial or basic requirements for the system. The analyst then quickly builds a prototype. When the prototype is completed, the users work with it and tell the analyst what they like and do not like about it. The analyst uses this feedback to improve the prototype and takes the new vet$ion back to the users. This interactive process continues until the users are relatively satisfied with what they have seen. Two key advantages of the prototyping technique are the large extent to which prototyping involves the user in a1nalysis and cksign and its ability to capture requirements in concrete, rather than verbal or abstract, from. In addition to being used as a stand-alone process, prototyping may also be used to augment the SDLC.

For example, a prototype of the final system may be developed early in analysis to help the analysts identify what users want. Then the final system is developed based on the specifications of the prototype.

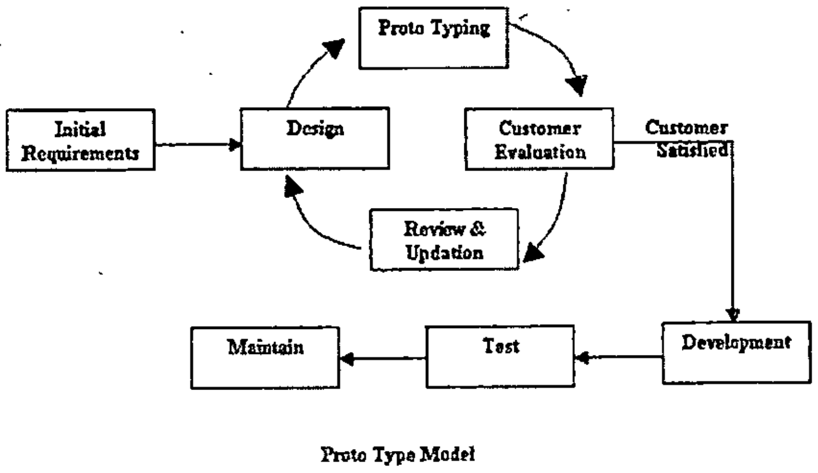


Fig. 15. The prototyping methodology.

CASE Tools

Other efforts to improve the systems development process have taken advantage of the benefits offered by computing technology itself. The result has been the creation and fairly widespread use of computer-aided software engineerin1, or CASE, tools.

CASE tools have been developed for internal use and for sale by several leading firms, including Oracle (Designer), Computer Associates (Advantage Gen), and IBM (Rational Rosè).

CASE tools are built around a central repository for system descriptions and specifications, including information about data names, format, uses, and locations. The idea of a central repository of information about a project is not new-the manual form of such a repo-sitory is called project dictionary or workbook. The difference is that CASE tools automate the repository for easier updating and consistency.

CASE tools also include diagramming tools for data flow diagrams and other graphical aids, screen and report design tools, and other special-purpose tools. CASE helps programmers and analysts do their jobs more efficiently and more effectively by automating routine tasks. In some organizations, CASE has been extremely successful, whereas in others it has not.

Case Support in Software Life Cycle

There are various types of support that CASE provides during the different phases of a software life cycle.

1. Prototyping Support

The prototyping is useful to understand the requirements of complex software products, to market new ideas and so on. The prototyping CASE tools requirements are as follows:

(i) Define user interaction

(ii) Define the system control flow

(iii) Store and retrieve data required by the system

(iv) Incorporate some processing logic.

Few features, which are supported by prototyping tools, are:

- Main use of prototyping CASE tool is developing Graphical User Interface (GUI) development. The user should be allowed to define all data entry forms, menus and control.
- Integrate well with the data dictionary of a CASE environment.
- It should be able to integrate with the external user-defined modules written in high-level languages.
- The user should be able to define the sequence of states through which a created prototype can run.
- The prototype should support mock up run of the actual system and management of the input and output data.

2. Structured Analysis and Design

A CASE tool should support one or more of the structured analysis and design techniques. It should also support making of the fairly complex diagrams and preferably through a hierarchy of levels. The tool must also check the incompleteness, inconsistencies and anomalies across the design and analysis through all levels of analysis hierarchy.

Analysis and design tools enable a software engineer to create models of the system to be built. The models contain a representation of data, function, and behaviour (at the analysis level) and characterizations of data, architectural, component level, and interface design.

By performing consistency and validity checking on the models, analysis and design tools provide a software engineer with some degree of insight into the analysis representation and help to eliminate errors before they propagate into the design, or worse, into implementation itself.

3. Code Generation

A support expected from a CASE tool during the code generation phase comprises the following:

- The CASE tool should support generation of module skeletons or templates in one or more popular programming languages.
- The tool should generate records, structures, class definitions automatically from the contents of the data dictionary in one or more popular programming languages..
- It should be able to generate database tables for relational database management system.
- The tools should generate code for user interface from prototype definitions for X-Windows and MS Window based applications.

4. Test CASE Generator

The CASE tool for test case generator should have following features:

- It should support both design and requirement testing.
- It should generate test set reports in ASCII format, which can be directly imported into the test plan document.

Under testing phase, test management tools are used to control and coordinate software testing for each of the major testing steps. Testing tools manage and coordinate regression testing, perform comparisons that ascertain differences between actual and expected output, and conduct batch testing of programs with interactive human/computer interfaces.

In addition to the functions noted, many test management tools also serve as generic test drivers. A test driver reads one or more test cases from a testing file, formats the test data to conform to the needs of the software under test, and then invokes the software to be tested.

Objective of Case

1. Improve Productivity: Most organizations use CASE to increase the speeds with which systems are designed and developed. Imagine the difficulties; the carpenters would face without hammers and saws. Tools increase the analysts' productivity by reducing the time needed to document, analyze, and construct; information system.
2. Improve Information System Quality: When tools improve processes, they usually improve the results as well.
    - Ease and improve the testing process through the use of automated checking.
    - Improve the integration of development activities via common methodologies.
    - Improve the quality and completeness of documentation.
    - Help standardize the development process.
    - Improve the management of the project.
    - Simplify program maintenance.
    - Promote reversibility of modules and documentation.
    - Shortens the overall construction process.
    - Improve software portability across environments.
    - Through reverse engineering and re-engineering, CASE products extend the file of existing systems.

    Despite the various driving forces (objectives) for the adoption of CASE, there are many resisting forces also that preclude many organizations from making investment in CASE.
3. Improve Effectiveness: Effectiveness means doing the right task (i.e., deciding the best task to perform to achieve the desired result). Tools can suggest procedures (the right way) to approach a task. Identifying user requirements, stating them in an

understandable form, and communicating them to all interested parties can be an effective development process compared to moving quickly into coding.

4. Organizations Reject CASE
   - The start-up cost of purchasing and using CASE
   - The high cost of training personnel
   - The big benefits of using CASE come in the late stages of the SDLC
   - CASE often lengthens the duration of early stage of the project
   - CASE tools cannot easily share information between tools
   - Lack of methodology standards within organizations, CASE products forces analysts to follow a specific methodology for system development
   - Lack of confidence in CASE products
   - Is personnel view CASE as a threat to their job security.

   Despite these issues, in long-term, CASE is very good. The functionality of CASE tools is increasing and the costs are coming down. During the next several years, CASE technologies and the market for CASE will begin to mature.

Case Repository

A CASE repository is a system developer database. Synonyms include dictionary and encyclopedia. It is a place where developers can store system models, detailed descriptions and specifications, and other products of system development.

Analysts use CASE repositories for five important reasons:

- To manage the details in large systems
- To communicate a common meaning for all system elements
- To document the features of the system
- To facilitate analysis of the details in order to evaluate characteristics and determine where system changes should be made.
- To locate errors and omissions in the system.

To limit the amount of narrative needed to describe relationships between data items and at the same time to show the structural relationship clearly, analysts often use formal notation in data dictionary, a component of CASE repository.

Data dictionary can be developed manually or using automated systems. Automated systems offer the advantage of automatically producing data element, data structure, and process listings; they also perform cross-reference checking and error detection. The data dictionary is a repository of all data definitions for all organizational applications and is used to manage and control access to the information repository, another component of CASE repository. Information repository provides automated tools used to manage and control access to business information and application portfolio.

CASE repository is an idea central to I-CASE. Integrated-CASE tools rely on common terminology, notations and methods for systems development across all tools. Within an I-CASE environment, all diagrams, forms, reports and programs can be automatically updated by the single change to the data-dictionary definition. Besid es specific tool integration, there are two additional ad vantages of using a compṕehensive CASE repository that relate to project management and reusability.

The CASE repository provides a wealth of information to the project manager and allows the manager to exert an appropriate amount of control on the project. If all organizational systems were created using CASE technology with a common repository, it would be possible to reuse significant portions of prior systems in the development of new ones.

Characteristics of Case Tools

All CASE tools have the following characteristics:

1. A graphic interface to draw diagrams, charts, models (upper case, middle case, lower case)
2. An information repository, a data dictionary for efficient information management selection, usage, application and storage
3. Common user interface for integration of multiple tools used in various phase
4. Automatic code generators
5. Automatic testing . tools.

Case Classification

Case classifications help us understand the different types of CASE tools aild their role in supporting software process activities. There are various different ways of classifying CASE tools, each of which gives us a different perspective on these tools. In this section, discuss CASE tools from three of these perspectives, namely:

1. A functional perspective where CASE tools are classified according to their specific function.
2. A process perspective where tools are classified according to the process activities which they support.
3. An integration perspective here CASE tools are classified according to how they are organized into integratėd units which provide support for one or more process activities.

Categories of Case Tools

The schematic diagram of CASE tools is drawn below in Fig. 16.

Notes

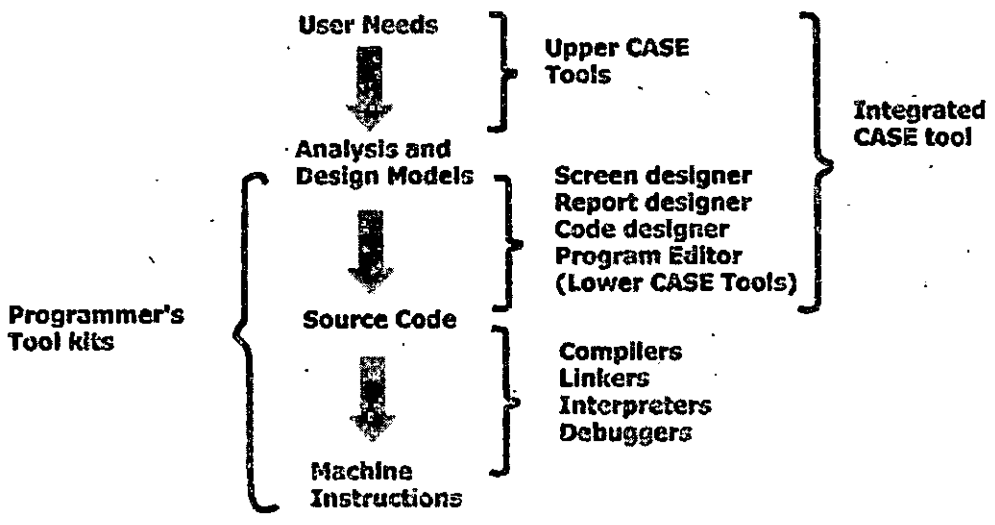


Fig. 16. Categories of CASE Tools

Smith and Oman have defined CASE tools which are divided into the following two categories.

- Vertical CASE tools
- Horizontal CASE tools

Vertical CASE Tools

Vertical CASE tools provide support for certain activities within a single phase of the software life cycle. There are two subcategories of vertical CASE tools:

(i) First Category. It is the set of tools that are within one phase of life cycle. These tools are important so that development in each phase can be as quick as possible.

(ii) Second Category. It is a tool that is used in more than one phase, but does not support moving from one phase to the next. These tools ensure that the developer does move on the next phase as appropriate.

Horizontal CASE Tools

These tools support automated transfer of information between the phases of a life cycle. These tools include project management, configuration management tools and integration services.

The above two categories of CASE tools can further be broken down into the following:

1. Upper CASE Tools/Front-End CASE Tools: CASE tools are designed to support the analysis and design phases of SDLC. All the analysis, design and specification tools are front-end tools. These tools also include computer-aided diagramming tools oriented

towards a particular programming design methodology, more recently including object-oriented design. The general types of upper CASE tools are listed below:

- Diagramming Tools: Diagramming tools enable system process, data and control structures to be represented graphically. They strongly support analysis and documentation of application requirements.
- Form and Report Generator Tools: They support the creation of system forms and reports in order to show how systems will "look and feel" to users.
- Analysis Tools: Analysis tools enable automatic checking for incomplete, inconsistent, or incorrect specifications in diagrams, forms and reports.

2. Lower CASE or Back -End Tools: CASE tools designed to support the implementation and maintenance phases of SDLC. All the generator, translation and testing tools are back-end tools. The.general types of Lower CASE tools are:
   - Code Generators: Cde generators automate the preparation of computer software. Code generation is not yet perfect. Thus, the best generator will produce approximately 75 percent of the source code for an application. Hand coding is still necessary.
3. Cross Life Cycle CASE or Integrated Tools: CASE tools used to support activities that occur across multiple phases of the SDLC. While such tools include both front-end and back-end capabilities, they also facilitate design, management, and maintenance of code. In addition, they provide an efficient environment for the creation, storage, manipulation, and documentation of systems.
4. Reverse Engineering Tools: These tools build bridges from lower CASE tools to upper CASE tools. They help in the process of analyzing existing applications, performa and database code to create higher level representations of the code.

Advantages of Case Tools

The major benefits of using CASE tools include the following:

1. Improved productivity
2. Better documentation
3. Improved accuracy
4. Intangible benefits
5. Improved quality
6. Reduced lifetime maintenance
7. Opportunity to non-programmers
8. Reduced cost of software
9. Produce high.quality and consistent documents
10. Impact on the style of a working of company
11. Reduce the drudgery in a software engineer's work

Notes

12. Increase speed of processing
13. Easy to program software
14. Improved coordination among staff members who are working on a large software project
15. An increase in project control through better planning, monitoring and communication.

Disadvantages of Case Tools I

1. Purchasing of CASE tools is not an easy task. Its cost is very high. Due to this reason small software development firm do not invest in case tools.
2. Learning Curve: In general cases programmer productivity may fall in initial phase of implementation as user need time to learn this technology.
3. Tool Mix: It is important to make proper selection of case tools to get maximum benefit from the case tools, so wrong selection may lead to wrong result.

Limitations of Case Tools

The major limitations of using CASE tools include:

- Cost
- Learning Curve
- Tool Mix

Cost

Using CAS. tools is a very costly affair. In fact, most firms engaged in software development on a small scale do not invest in CASE tools because they think that the benefits of CASE are justifiable only in the development of large systems.

The cost of outfitting every system developer with a preferred CASE tool kit can be quite high. Hardware and systems, software, training and , consulting are all factors in the total cost equation of using CASE tools.

Learning Curve

A learning curve is a correlation between a learner's performance on a task and the number of attempts or time required to complete the task; this can be represented as a direct proportion on a graph. In most CASES, programmer productivity may fall in the initial phase of implementation, because users need time to learn the technlogy. The consultants offer training and on-site services that can be crucial to accelerate the learning curve and to the development and use of the tools.

Tool Mix

It is most important to make an appropriate selection of tool mix to get cost advantage. CASE integration and data integration across all platforms is also very important. The ability to share the results of work done on one CASE tool with another CASE tool is perhaps the most important type of CASE integratio.

1.17 INVESTIGATION

A system ismade to solve a problem. The process therefore has to start with recognition of the need. This step is called problem definition. In this phase the key question to be answered is "what is the problem that the system has to solve?"

This phase involves initial investigation and survey and should result in a clear statement of the scope and objectives of the system. It should give a clear idea of what is expected from the system in terms of performance criteria.

The first step in the system development life cycle requires the identification of a need. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such re.quests, standard procedu.tes must be established to deal with them. The initial investigation is one way of handling this. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system, or build a new one

Performing Initial Investigation

The first step in the system development life cycle requires the identification of a need. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such re.quests, standard procedu.tes must be established to deal with them. The initial investigation is one way of handling this.

The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system, or build a new one.

Problem Definition and Project Initiation

Initial investigation starts with the definitions of the problem. The problem ha.$ to be clearly stated. The user's request form specifies the following:

- Date request to be submitted.
- Date job should be completed.
- Job objective(s)-purpose of job requested.
- Expected benefits to be derived from proposed change.
- Input/output description-quantity (number of copies or pages) and frequency (daily, weekly, etc.) of inputs and outputs of proposed change..
- Requester's signature, title, department and phone number.
- Signature, title, department, and phone number of person approving the request.

The user request identifies the need for change and authorises the initial investigation. It may undergo several modif}cations before it becomes a written commitment. Once the request is approved, the following activities are carried out: background investigation, fact-finding and analysis, and presentation of results-called project proposal. The proposal, when approved, initiates a detailed user-oriented specification of system petfotmance and analysis of the feasibility of the candidate system.

Notes

A feasibility study focuses on identifying and evaluating alternative candidate systems with' a recommendation of the best sys.tern for the job. The user or the system analyst may identify the need for a candidate system or for enhancements in the existing system.

Often problems come into focus after a joint meeting between the user and the analyst. In either case, the user initiates an investigation by filling out a request form for information. The request provides for statements ofobjective and expected benefits.

Data and Fact Gathering Techniques

Fact finding is an important step for the analyst, based on which the analyst gains an understanding of the existing system and its problem, and of the requirement from the new system. There are a number of fact finding tools, which the analyst uses fur this; these are discussed in the following sub-section. The analyst chooses the tools he eonsiders suitable and uses them to gather all the required in:fotmattoni which is then analyz.ed.

Study Existing Documents and Records

Many systems and organizations have some type of documents and records already available. Examples of such documents are:

- Input form
- Existing system user manuals, and other system documentation
- System review/audit
- Correspondence, for example, problem logs, request for enhancements
- Brochures
- Reports from the system
- Data file procedure manuals.

Study of available documentation is a fast and person-independent way of fact gathering. The analyst can study these document comfortably and. use these to prepare further questions for the rest of the fact gathering exercise.

Interviews

- Personal interviews are a direct method of obtaining information from people. By this the analyst learns about the existing system, problems. and expectations.
- During an. interview the analyst and the person being interviewed are meeting face to face.This givsthe analyst a flexible opportunity to learn various pertinent facts. As the interviewer is facing the interviewee, he can observe the reaction of the interviewee and see how the person being asked a question is responding to it. This gives the interviewer an idea of the reliability of the information being gathered. It also gives the interviewer chance to ask more questions along with some promising line, which may not have been thought of earlier.

Questionnaires

- A questionnaire seeks information from persons in a written form and in a prescril;>e format against a set of questions. Questionnaires are good for gathering specific information where the questions can be structured in advance and uniformly for a number of persons.
- Questionnaires are useful as they are a quick means of gathering information and analyzingit together; They are particularly useful if the respondents are scattered geographically or there is no time to hold interviews.

Questions could be:

- Structured-here the respondent has to select from possible options and the range of answers is limited.
- Unstructured-asking the respondent's opining and letting the respondent answer freely. Such questions are open-ended.

Examples of structured questions are where the answer could be:

- Yes/no type
- Selected from specified multiple choice
- Selections on a ranking scale
- Selections of a rating
- Fill in the blank.

Structured questions need more analysis and may or may not give the type of information sought. Also, analyzing them may introduce . analyst bias. They may however give better insights into the problems as they are open ended and exploratory. Often, they need follow up in the form of interviews.

Group Communication

When information is required from face to-face sessions, but there is not enough time to conduct personal interviews, meetings can be held. Here, since there are many persons present, more types ofideas can be discussed in a short time. Also, the comments of one person may prompt other persons to contribute facts they have otherwise forgotten.

Conducting a group session is a skilled matter as there are problems like:

- The group may be dominated by some persons and others, who may otherwise have something to contribute, may be shy and not speak up.
- Problems with seniors may not be voiced because of seniors being present.
- The situation could lead to a verbal fight between persons which may need moderation.
- Internal politics of an organization may determine what is said and what. is left unsaid, thus resulting in a false picture.
- There may be situations where may persons talk simultaneously and are agitated and the comments they make may not be heard or get recorded.

Notes

Presentations

At times, the analyst may hold a presentation presenting his understanding of the system and problems etc. Such a presentation would typically involve showing slides and talking to a group of users whose response is sought. The persons attending the presentation have an opportunity to respond and confirm of affect the analyst's understanding.

Presentations are useful in situations where the users are passive or too busy to actively explain things. Here, an analyst may use study of existing records and questionnaires etc., to put together a presentation

Fact Analysis

Data gathered by analyst has to be organized and evaluated to draw conclusions. Vous means used to document and anlyze the data gathered include:

- Flow charts
- DFDs
- Decision tables
- Structure charts.

Analysis is done using techniques like data elements:

- Input-output
- Recurring data
- Reports usage.

Concluding the Initial Investigation

By this stage, the analyst has a thorough knowledge of the system as part of the initial investigation. The initial investigation culminates with a system proposal or a project request;

Work on the initial investigation was initiated with a system request or an information system request which gave the system objective, benefit expected, output descriptions, input descriptions. This document is modified as a result of the initial investigation,; even the objectives may be modified and refined. The benefits and the input and output descriptions are expanded. The requirements for the feasibility study resources are also included.

The modified system request is presented by the analyst and is reviewed by the user. Review comments of the user are then incorporated to generate a final document.

1.18 IMPLEMENTATION

Implementation is the process of first ensuring that the information system is operational and then allowing users to take over its operation for use and evaluation. This involves training the users to handle the system. The analyst needs to plan for a smooth conversion from the old system to the new one. This includes converting files old formats to new ones or building new databases etc.

Once the Information System has been developed and acceptance testing is completed, the implementation process starts. Users must be trained on the use of the new system, focusing on its requirements and its capabilities. Many organisations combine testing and training in the same stage. This works well because users can -become familiar with the new is as well as ensure that it can handle errors at the same time. Training, like testing and documentation, is ultimately a management responsibility

Software Implementation Guidelines

Software implemeniation should be done with proper planning, hasty decisions lead to problems and delays. The process of implementation of software should be consistent with least amount of disturbance. Following are the some of the basic considerations that should be kept in mind for smooth implementation of software.

Proper Equipment

The hardware and software requirements should be first re-examined with the software supplier. The equipments for general-purpose applications should be delivered several weeks before the installation of an application. Thes helps to have a basic idea about the hardware before major implementation of the applications. Sufficient time for networking should be given, if the system uses the network resources. Good quality wiring and the fastest hubs should be used to gain best performance from the system. Proper connectivity devices such as wiring, hubs and routers should be used to reduce bottleneck arising in system performance.

Conversion

Conversion methodology ensures a professional result in line with expectations and within budgeted time period and cost. The conversion methodology clearly improves communication between the project team arid management by providing a readily understandable, structured approach.

Training

Formal training about the functioning of software should be provided to the employees for the successful use of the application software. Hardly eyer, one reads a manual and implements the application in a smooth manner. Application training should be designed to teach users how to use the software.

Implementing Applications

Follow a definite sequence to install all the inter-related applications. For example, an application consists of general accounting, payroll and utility billing. In the application, accounting will come first because the other systems require its availability.

Backups

Regular system backups should be taken so that the users can revert to an older copy of the data files when they commit any mistake. The backup procedure can be performed

Notes

during the free hours of at the end of the day. You can take the take the backup of only the critical points in an application. Backups are stored in the removable disk or in high capacity tapes. You can also store backups in another folder 'of the hard disk drive of your system.

1.19 RELATION BETWEEN DESIGN AND IMPLEMENTATION

System Design

Design is the most creative and challenging phase of the system development life cycle. The term design describes the final system and process by which it is developed. Different stages of design phase are shown in Figl. This phase is very important phase of life cycle. This is a creative as well as a technical activity including the following tasks:

- Appraising the terms of reference
- Appraising the analysis of the existing system, particularly regarding problem areas
- Defining precisely the required system output
- Determining data required to produce the output
- Deciding the medium and open the files
- Devising processing methods and use of software to handle files and to produce output
- Determining methods of data capture and data input
- Designing the output forms
- Difining detailed critical procedures
- Calculating timings of processing and data movements
- Documenting all aspects of design

Implementation

Implementation phase is less creative than system design. It is mainly concerned with user training, site selection preparation and file conversion. Once the system has been designed, it is ready for implementation. Implemention is concerned with those tasks leading immediately to a fully operational system. It involves programmers, users and operations management; but it's planning and timing is a prime function of system's analyst. Itincludes the final testing of complete system to user satisfaction, and supervision of initial operation of the system. Implementation of the system includes providing security to the system, also so that some person may not misuse it.

Types of Implementation

There are three types of implementation:

- Implementation of a computer system to replace a manual system.

- Implementation of a new computer system to replace an existing one.
- Implementation of a modified application (software) to replace an existing one using the same computer

Coding

Good software development organiz ations normally require their programm.ers to adhere to some we-defined and standard style of coding called coding standards. Most software development organizations formulate their own coding standards that suit them most, and require their engineers to follow these standards rigorously. The purpose of requiring all engineers of an organization to adhere to a standard style of coding is the following:

- A coding standard gives a uniform appearance to the codes written by different engineers.
- It enhances code understanding.
- It encourages good programming practices.

A coding standard lists several rules to be followed during coding, such as the way variables are to be named, the way the code is to be laid out, error return conventions, etc

Coding Standards and Guidelines

Good software development organizations usually develop their own coding standards and guidelines depending on what best suits their organization and the type of products they develop. The following are some representative coding standards.

Rules for limiting the use of global: These rules list what types of data can be declared global and what cannot.

Contents of the headers preceding codes for different modules: The information contained in the headers of different modules should be standard for an organization. The exact format in which the header infoımation is organized in the header can also be specified. The following are some standard header data:

- Name of the module .
- Date on which. the module was created. ',
- Author's name.
- Modification history.
- Synopsis of the module.
- Different functions supported, along with their input/output parameters.
- Global variables accessed/modified by the module.

Naming conventions for global variables, local variables, and constant identifiers: A possible naming convention can be that global variable names always start with a capital letter, local variable names are made of small letters, and constant names are always capital letters.

Error return conventions and exception handling mechanisms:

The way error conditions are repotted by different functions i a program are handled

Notes

should be standard within an organization. For example, different functions while encountering an error condition should either return a 0 or 1 consistently.

The following are some representative coding guidelines recommended by many software development organizations.

- Do not use a coding style that is too clever or too difficult to understand: Code should be easy to understand. Many inexperienced engineers actually take pride in writing cryp_tic and incomprehensible code. Clever coding can obscure meaning of the code and hamper understanding. It also makes maintenance difficult.
- Avoid obscure side effects: The side effects of a function call include modification of parameters passed by reference, modification of global variables, and 1/0 operations. An obscure side effect is one that is not obvious from a casual examination of the code. Obscure side effects make it difficult to understand a piece of code. For example, if a global variable is changed obscurely in a called module or some file 1/0 is performed which is difficult to infer from the function's name and header information, it becomes difficult for anybody trying to understnd the code.
- Do not use an identifier for multiple purposes: Programmers often use the same identifier to denote several temporary entities. For example, some programmers use a temporary loop variable for computing and a storing the final result. The rationale that is usually given by these programmers for such multiple uses of variables is memory efficiency, e.g., three variables use up three memory locations, whereas the same variable used in three different ways uses just one memory location. However, there are several things wrong with this approach and hence should be avoided. Some of the problems caused by use of variables for multiple purposes as follows:
- Each variable should be given a descriptive name indicating its purpose. This is not possible if an identifier is used for multiple purposes. Use of a variable for multiple purposes can lead to confusion and make it difficult for somebody trying to read and understand the code.
- Use of variables for multiple purposes usually makes future enhancements more difficult.

The code should be well-documented: As a rule of thumb, there must be at least one comment line on the average for every three source line. The length of any function should not exceed 10 source lines: A function that is very lengthy is usually very difficult to understand as it probably carries out many different functions. For the same reason, lengthy functions are likely to have disproportionately larger number of bugs.

Do not use go to statements: Use of go to statements makes a program unstructured and makes it very difficult to understand.

Code Review

Code review for a model is carried out after the module is successfully compiled and the all the syntax errors have been eliminated. Code reviews are extremely cost-effective strategies for reduction in coding errors and to produce high quality code. Normally, two types of reviews are carried out on the code of a module. These two types code review techniques are code inspection and code walk through.

Code Walk Through

Code walk through is an informal code analysis technique. In this technique, after a module has been coded, successfully compiled and all syntax errors eliminated. A few members of the development team are given the code few days before the walk through meeting to read and understand code. Each member selects some test cases and simulates execution of the code by hand (i.e., trace execution through each statement and function execution). The main objectives of the walk through are to discover the algorithmic and logical errors in the code. The members note down their findings to discuss these in a walk through meeting where the coder of the module is present.

Even though a code walk through is an informal analysis technique, several guidelines have evolved over the years for making this naive but useful analysis technique more effective. Of course, these guidelines are based on personal experience, common sense, and several subjective factors. Therefore, these guidelines should be considered as examples rather than accepted as rules to be applied dogmatically.

Some of these guidelines are the following.

- The team performing code walk through should not be either too big or too small. Ideally, it should consist of between three to seven members.
- Discussion should focus on discovery of errors and not on how to fix the discovered errors.
- In order to foster cooperation and to avoid the feeling among engineers that they are being evaluated in the code walk through meeting, managers should not attend the walk through meetings

Code Inspection

In contrast to code walk through, the aim of code inspection is to discover some common types of errors caused due to oversight and improper programming. In other words , during code inspection the code is examined for the presence of certain kinds of errors, in contrast to the hand simulation of code execution done in code walk throughs. For instance, consider the classical error of writing a procedure that modifies a formal parameter while the calling routine calls that procedure with a constant actual parameter.

It is more likely that such an error will be discovered by looking for these kinds of mistakes in the code, rather than by simply hand simulating execution of the procedure. In addition to the commonly made errors, adherence to coding standards is also checked during code inspection. Good software development companies collect statistics regarding different types of errors commonly committed by their engineers and identify the type of errors most frequently committed. Such a list of commonly committed errors can be used during code inspection to look out for possible errors.

Following is a list of some classical programming errors which can be checked during code inspection:

1. Use of uninitialized variables.
2. Jumps into loops.

3. Nonterminating loops.
4. Incompatible assignments.
5. Array indices out of bounds.
6. Improper storage allocation and deallocation.
7. Mismatches between actual and formal parameter in procedure calls.
8. Use of incorrect logical operators or incorrect precedence among operators.
9. Improper modification of loop variables.
10. Comparison of equally of floating point variables, etc.

Clean Room Testing

Clean room testing was pioneered by IBM. This type of testing relies heavily on walk throughs, inspection, and formal verification . The programmers are not allowed to test any of their code by executing the code other than doing some syntax testing using a compiler. The software development philosophy is based on avoiding software defects by using a rigorous inspection process. The objective of this software is zero-defect software.

The name 'clean room' was derived from the analogy with semiconductor fabrication units. In these units (clean rooms), defects are avoided by manufacturing in ultra-clean atmosphere. In this kind of development, inspections to check the consistency of the components with their. specifications has replaced unit-testing.

This technique reportedly produces documentation and code that is more reliable and maintainable than other development methods relying heavily on code execution-hased testing.

The clean room approach to software development is based on five characteristics:

- Formal specification: The software to be developed is formally specified. A state-transition model which shows system responses to stimuli is used to express the specification.
- Incremental development: The software is partitioned into increments which are developed and validated separately using the clean room process. These increments are specified, with customer input, at an early stage in the process.
- Structured programming: Only a limited number of control and data abstraction constructs are used. The program development process is process of stepwise refinement of the specification
- Static verification: The developed software is -statically verified using rigorous software inspections. There is no unit or module testing process for code components.
- Statistical testing of the system: The integrated softwarr increment is tested statistically to determine its reliability. These statistical tests are based on the operational profile which is developed in parallel with the system specification.
- The main problem with this approach is that testing effort is increased as walk through, inspection, and verification are time-consuming.

Software Documentation

When various kinds of software products are developed then not only the executable files and the source code are developed but also various kinds of documents such as users' manual, Software Requirements Specification (SRS} documents, design documents, test documents, installation manual, etc., are also developed as part of any software engineering process. All these documents are a vital part of good software development practice. Good documents are very useful and server the following purposes:

- Good documents enhance understandability and maintainability of a software product. They reduce the effort and time required for maintenance.
- Use documents help the users in effectively using the system.
- Good documents help in effectively handling the manpower turnover problem. Even when an engineer leaves the organization, and a new engineer comes in, he can build up the required knowledge easily.
- Production of good documents helps the manager in effectively tracking the progress of the project. The project manager knows that measurable progress is achieved if a piece of work is done and the required documents have been produced and reviewed.

Different types of software documents can .broadly be classified into the following:

- Internal documentation
- External documentation

Internal documentation: is the code comprehension features provided as part of the source code itself. Internal documentation is provided through appropriate module headers and comments embedded in the source code. Internal documentation is also provided through the useful variable names, module and function headers, code indentation, code structuring, use of enumerated types and constant identifiers, use of user-defined data types, etc. Careful experiments.

Suggest that out of all types of internal documentation meaningful variable names is most useful in understanding the code. This is of course in contrast to the common expectation that code commenting would be the most useful. The research finding is obviously true when comments are written without thought. For example, the following style of code commenting does not in any way help in understanding the code.

```
a = 10;/* a made 10*/
```

·But even when code is carefully commented, meaningful variable names still are more helpful in understanding a piece of code. Good software development organizations usually ensure good internal documentation by appropriately formulating their coding standards and coding guidelines.

External documentation: is provided through various types of supporting documents such as users' manual, software requirements specification document, design document, test documents, etc. A systematic oftware development style ensures that all these documents are produced in an orderly fashion.

Notes

1.20 MAINTENANCE

It is a task that every development team has to face when the software is delivered to the customer's site, installed and is operational. In general, it means fixing things that breaks out or wear out. In software, nothing wears out it is either wrong from the beginning or we decide later that we want to do something different.

It is a very broad activity that includes error corrections, enhancements of capabilities, deletion of obsolete capabilities and optimization. Because change is inevitable, mechanisms, must be developed for evaluating, controlling and making modifications. So any work done to change the software after it is in operation is considered to be maintenance work.

Software maintenance is becoming an important activity of a large number of software organizations. There is no surprise, given the rate of hardware obsolescence, the immortality of a software product and the demand of the user community to see the existing software products run on newer platforms, run in newer environments, and/or with enhanced features. When the hardware platform is changed, and a software product performs some low-level functions, maintenance is necessary.

Also, whenever the support environment of a software product changes, the software product requires rework to cope up with the newer interface. For instance, a software product may need to be maintained when the operating system changes.

Thus, every software product continues to evolve after its development through maintenance efforts. Therefore, it can be stated that software maintenance is needed to correct errors, enhance features, port the software to new platforms, etc

Types of Software Maintenance

There are basically three types of software maintenance. These are:

- Corrective: Corrective maintenance of a software product is necessary to rectify the bugs observed while the system is. in use.
- Adaptive: A software product might need maintenance when the customers need the product to run on new platforms, on new operating systems, or when they need the product to interface with new hardware or software.
- Perfective: A software product needs maintenance to support the new features that users want it to support, to change different functionalities of the system according to customer demands, or to enhance the performance of the system

Problmes associated with Software Maintenance

Software maintenance work typically is much more expensive than what it should be and takes more time than required. In software organizations, maintenance work is mostly carried out using ad hoc techniques. The primary reason being that software maintenance is one of the most neglected areas of software engineering. Even though software maintenance is fast becoming an important area of work for many companies as the software products of yester years age, still software maintenance is mostly being carried out as fire-fighting operations, rather than through systematic and planned activities. Software maintenance has . very poor image in industry.

Therefore, an organization often cannot employ. bright engineers to carry out maintenance work. Even though maintenance suffers from a poor image, the work involved is often more challenging than development work. During maintenance it is necessary to thoroughly understand someone else's work and then carry out the required modifications and extensions.

Another problem associated with maintenance work is that the majority of software products needing maintenance are legacy products.

Software Reverse Engineering

Software reverse engineering is the process of recovering the design and the requirements specification of a product from an analysis of its code. The purpose of reverse engineering is to facilitate maintenance work by improving the understandability of a system and to produce the necessary documents for a legacy system. Reverse engineering is becoming important, since, legacy software prod ucts lack proper documentation, and are highly unstructured. Even well-designed products become legacy software as their structure degrades through a series of maintenance efforts.

The first stage of reverse engineering usually focuses on carrying out cosmetic changes to the code to improve its readability, structure, and understandability, without changing of its. functionalities. A process model fo:r reverse engineering has been shown in Fig. l(a). A program can be reformatted using any of the several available pretty printer programs which layout the program neatly. Many legacy software products with complex control structure and un thoughtful variable names are difficult to comprehend. Assigning meaningful variable names is important because meaningful variable namesare the most helpful thing in code documentation. All variables, data structures, and functions should be assigned meaningful names wherever possible . Complex nested conditionals in the program can be replaced by simpler conditional statements or whenever appropriate by case statements.

After the cosmetic changes have been carried out on a legacy software, the process of extracting the code, design, and the requirements specification can begin. These activities are schematically shown in Fig.2. In order to extract the design, a full understanding of the code is needed. Some • automatic tools can be used to derive the data flow and control flow diagram from the code. The structure chart (module invocation sequence and data interchange among mod ules) should also be extracted. The SRS document can be written once the full code has been thoroughly understood and the design extracted.

Legacy Software Products

It is prudent to define a legacy system as any software system that is hard to maintain. The typical problems associated with legacy systems are poor documentation, unstructured (spaghetti code with ugly control structure), and lack of personnel knowledgeable in the• product. Many of the legacy systems were developed long time back. But, it is possible that a recently developed system having poor design and documentation can be considered to be a legacy system.

Notes

Factors on which Software Maintenance Activities Depend

The activities involved in a software maintenance project are not unique and depend on several factors such as:

- the extent of modification to the product required
- the resources available to the maintenance team
- the conditions of the existing product (e.g., how structured it is, how well documented it is, etc.)
- the expected project risks, etc.

When the changes needed to a software product are minor and straightforward, the code can be directly modified and the changes appropriately reflected in all the documents. But more elaborate activities are required when the required changes are not so trivial. Usually, for complex maintenance projects for legacy systems, the software process can be represented by a reverse engineering cycle followed by a forward engineering cycle with an emphasis on as much reuse as possible from the existing code and other documents.

Software Maintenance Process Models

Two broad categories of process models for software maintenance can be proposed. The first model is preferred for projects involving small reworks where the code is changed directly and the changes are reflected in the relevant documents later. This maintenance process is graphically presented in Fig.3.

In this approach, the project starts by gathering the requirements for changes. The requirements are next analyzed to formulate the strategies to be adopted for code change. At this stage, the association of at least a few members of the original development team goes a long way in reducing the cycle team, especially for projects involving unstructured and inadequately documented code. The availability of a working old system to the maintenance engineers at the maintenance site greatly facilitates the task of the maintenance team as they get a good insight into the working of the old system and also can compare the working of their modified system with the old system. Also, debugging of the re engineered system becomes easier as the program traces of both the systems can be compared to localize the bugs.

The second process model for software maintenance is preferred foı projects where the amount of rework required is significant. This approach can be represented by a reverse engineering cycle followed by a forward engineering cycle. Such an approach is also known as software re engineering. This process model is depicted in Fig. 17. The reverse engineering cycle is required for legacy products. During the reverse engineering, the old code is analyzed (abstracted) to extract the module specifications.

The module specifications are then analyzed to produce the design. The design is analyzed (abstracted) to produce the original requirements specification . The change requests are then applied to this requirements specification to arrive at tlie new requirements specification. At the design; module specification, and coding a substantial reuse is made from the reverse engineered products. An important advantage of this approach is that it produces a more structured design compared to what the original product had, produces good documentation, and very often results in increased efficiency. The efficiency improvements are brought about by a more efficient design. However, this approach is more costly than the first approach.

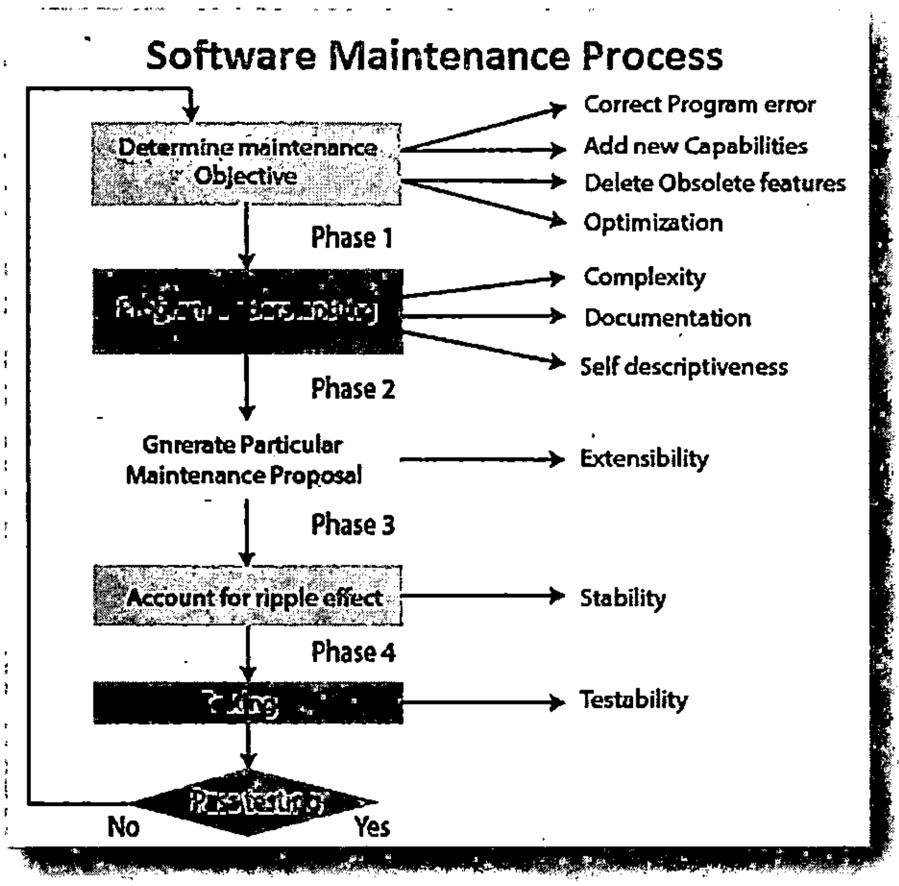


Fig. 17 Maintenance process model 1

An empirical study indicates that process 1 is preferable when the amount of rework is no more than 15%. Besides the amount of rework, several other factors might affect the decision regarding using process model 1 over process model 2:

- Re-engineering might be preferable for products which exhibit a high failure rate.
- Re-engineering might also he preferable for legacy products having poor design and code structure.

Estimation of Approximate Maintenance Cost

It is well known that maintenance efforts require about 60% of the total life cycle cost for a typical software product. However, maintenance costs vary widely from one application domain to another. For embedded systems, the maintenance cost can be as much as 2 to 4 times the development cost. Boehm [1981] proposed a formula for estimating maintenance costs as part of his COCOMO cost estimation módel. Boehm's maintenance cost estimation

Notes

is made in terms of a quantity called the Annual Change Traffic (ACT).

Boehm defined ACT as the fraction of a software product's source instructions which undergo change during a typical year either through addition or deletion.

$$ACT = KLOC_{added} + KLOC_{deleted} / KLOC_{total}$$

Where, KLOC added is the total kilo lines of source code added during maintenance. KLOC deleted is the total KLOC deleted during maintenance. Thus, the code that is changed, should be counted in both the code added and the code deleted. The Annual Change Traffic (ACT) is multiplied with the total development cost to arrive at the maintenance cost: maintenance cost = ACT x development cost.

Most maintenance cost estimation models, however, yield only approximate results because they do not take into account several factors such as experience level of the engineers, and familiarity of the engineers with the product, hardware requirements, software complexity, etc.

Documentation

Software documentation is the written of facts about a software system recorded with the intent to convey purpose, content and clarity. The recording process usually begins when the need for the system is conceived and continues until the system is no longer in use.

Software documentation is a written piece of text that is often accompanied with a software program. This makes the life of all the members associated with the project more easy. It may contain anything from API documentation, build notes or just help content. It is a very critical process in software development. It's primarily an integral part of any computer code development method.

Moreover, the computer code practitioners are a unit typically concerned with the worth, degree of usage and quality of the actual documentation throughout development and its maintenance throughout the total method. Motivated by the requirements of Novatel opposition, a world leading company developing package in support of worldwide navigation satellite system, and based mostly on the results of a former systematic mapping studies area unit aimed of the higher understanding of the usage and therefore the quality of a varied technical documents throughout computer code development and there maintenance.

For example before development of any software product requirements are documented which is called as Software Requirement Specification (SRS). Requirement gathering is considered as an stage of Software Development Life Cycle (SDLC).

Another example can be a user manual that a user refer for installing, using, and providing maintenance to the software application/product.

Types Of Software Documentation :

- **Requirement Documentation:** It is the description of how the software shall perform and which environment setup would be appropriate to have the best out of it. These are generated while the software is under development and is supplied to the tester groups too.
- **Architectural Documentation:** Architecture documentation is a special type of

documentation that concerns the design. It contains very little code and is more focused on the components of the system, their roles and working. It also shows the data flows throughout the system.

- **Technical Documentation:** These contain the technical aspects of the software like API, algorithms etc. It is prepared mostly for the software devs.
- **End-user Documentation:** As the name suggests these are made for the end user. It contains support resources for the end user.
- **Purpose of Documentation:** Due to the growing importance of the computer code necessities, the method of crucial them needs to be effective so as to notice desired results. As, such determination of necessities is often beneath sure regulation and pointers that area unit core in getting a given goal.

These all implies that computer code necessities area unit expected to alter thanks to the ever ever-changing technology within the world . however the very fact that computer code information id obtained through development has to be modified within the wants of users and the transformation of the atmosphere area unit inevitable.

What is more, computer code necessities ensures that there's verification and therefore the testing method , in conjunction with prototyping and conferences there are focus teams and observations.

For a software engineer reliable documentation is often a should the presences of documentation helps keep track of all aspects of associate application and it improves on the standard of a wares, it's the most focuses area unit development , maintenance and information transfer to alternative developers. productive documentation can build info simply accessible, offer a restricted range of user entry purpose , facilitate new user learn quickly , alter the merchandise and facilitate to chop out the price.

Importance of software documentation

For a programmer reliable documentation is always a must the presence keeps track of all aspects of an application and helps in keeping the software updated.

Advantage of software documentation

- The presence of documentation helps in keeping the track of all aspects of an application and also improves on the quality of the software product .
- The main focus are based on the development , maintenance and knowledge transfer to other developers.
- Helps development teams during development.
- Helps end-users in using the product.
- Improves overall quality of software product
- It cuts down duplicative work.
- Makes easier to understand code.
- Helps in establishing internal co-ordination in work.

Notes

Disadvantage of software documentation :

- The documenting code is time consuming.
- The software development process often takes place under time pressure, due to which many times the documentation updates don't match the updated code .
- The documentation has no influence on the performance of the an application.
- Documenting is not so fun, its sometime boring to a certain extent.

The agile methodology encourages engineering groups to invariably concentrate on delivering price to their customers. This key should be thought-about within the method of manufacturing computer code documentation .good package ought to be provided whether or not it's a computer code specifications document for programmers , testers, computer code manual for finish users .

SUMMARY

- There are many system concepts which play an important role in understanding the system. The flow of information in an organization is very vital. There are various departments in an organization, depending on the services or products they provide to us. With each department there are three traditional levels of management-top, middle and lower. For making the proper decisions-the different levels of managers require the right kind. of information at right time. Information syatem is a system that provides information to people in an organization.
- Physical systems are tangible entities that may be static or dynamic in operation. For example, the physical parts of the computer center are the offices, desks, and chairs that facilitate operation of the computer. They can be seen and counted; they are static. In contrast, a programmed computer is a dynamic system. Data, programs, output, and applications change as the user's demands or the priority of the information requested changes.
- Environment represents everything external to a system that interacts with the system. Interface represents point of contact where a system meets its environment or where subsystems meet each other.
- Constraint is the limit to what a system can accomplish.
- Inp-ut is what.ever a system takes from its environment in order to fulfill its purpoae. Output is whatever a system returns to its environment in order to fulfill its purpose. Open system is a system that interacts freely with its environment, taking input and returning output. Closed systein is a system that is cut off from its environment and does not interact with it.
- Decomposition is the process of breaking down a system into smaller and less complex pieces that are easier to understand.
- Modularity means dividing a system up into chunks or modules of a relatively uniform size. Coupling is the extent to which a subsystems depend on each other.

- Cohesion is the extent to which a system or a subsystem performs a single function.
- Transaction Processing System (TPS) is the computer-based information system that keeps track of the transactions needed to conduct business.
- Management Information System (MIS) is the computer-based information system that derives data from all departments of an organization and produces summary, exception, periodic, and on demand reports of the organization's performance.
- Decision Support System (DSS) is the computer-based information system that helps managers vrith nonroutine decision-making tasks.
- Executive Support System (ESS) is also called an executive information system. It is made especially for top managers that specifically supports strategic decision making.
- Office Automation System (OAS) is the computer:.baaed information system that combines various technologies to reduce the manual labor needed to operate an office efficiently; used at all levels of an organization.
- Expert System is also called knowledge based system. It is a set of computer programs that perform a task at the level of a human expert. A system is defined as a collection of related components that interact to perform a task in order to accomplish a goal. Syatem analysis and design is a six-phase problem-solving procedure for examining an information system and improving it.
- The System Development Life Cycle (SDLC) is the step-by-step process that many organizations follow during systems analysis and design. The purpose of preliminary investigation is to conduct a preliminary analysis, propose alternative solutions, describe costs and benefits and submit a preliminary plan with recommendations.
- Data now Diagram (DFD) is a modelling tool that graphically shows the flow of data through a system.

KEY WORDS

- **Process:** set of interrelated or interacting activities which transforms inputs into outputs, uses resources to do it.
- **Software Process Model:** determine the order of the stages involved in software development and evolution and to establish the transition criteria for progressing from one stage to the next
- **Plan-driven development:** document-driven, code- driven, traditional emphasis on defining the scope, schedule, and costs of the project upfront extensive documentation (of product requirements), no iteration.
- **Change-driven development:** enabling the evolution of product requirements during the development, iterative and incremental.
- **Iterative development:** lifecycle model in which the software is built in iterations each iteration a mini-project requirements analysis, design, implementation and testing are conducted in order to produce a subset of the final system often resulting in internal iteration release (IIR)

Notes

- **Incremental development** - adding functionality to a system over several releases one incremental delivery may be composed of several iterations
- **IID – Iterative and Incremental Development** the software is built in iterations functionality is added to the system over several releases
- **PM – Project Management,** establish and carry out in a systematic way the tasks of the software implementation project, which allows complying with the project's objectives in the expected quality, time and cost
- **SI – Software Implementation** – the systematic performance of the analysis, software component identification, construction, integration and tests, and product delivery activities for new or modified software products according to the specified requirements

REVIEW QUESTIONS

1. What is the difference between a logical system description and physical system description ?
2. What are the departments, tasks and levels of managers in an organization, and what type of decisions do they make?
3. What are the different types of maintenance?
4. Give an example of a system around you and identify its characteristics.
5. Describe Development life cycle.
6. What are the elements of system? Explain the characteristics of a system.
7. Briefly explain the types of systems.
8. What are the disadvantages associated with software maintenance?
9. Explain the software implementation guidelines.
10. What are the variations n SDLC model? Explain

FURTHER READINGS

1. Software Engineering, Bharat Bhushan Agarwal, Sumit Prakash Tayal, Firewall Media.
2. Softwaree Engineering, D. Sunder, University Science Press
3. Kaniclides, A., C. Kimble.(1995) A Development Framework for Executive Information Systems. Proceedings of GRONICS'95, Groningen, The Netherlands, February.
4. Kimble, C., (1997) Assessing the Relative Importance of Factors Affecting Information Systems Success, University of York Technical Report Series, YCS 283, Department of Computer Science, York, UK.

Unit 2

System Planning and Investigation

Notes

Structure

2.0 LEARNING OBJECTIVES

After reading this chapter students will be able to:

- know about the systems planning and investigation
- discuss the basis for planning in systems analysis
- understand the dimensions of planning
- describe the initial investigation
- know the determining the user's information requirements
- explain the feasibility study and considerations
- describe the steps in feasibility analysis and feasibility report

2.1 INTRODUCTION

Planning information systems has become increasingly important because information is a vital reimurce and company asset. more and 'more funds are committed to information systems, and system development is a serious business for computers that incorporate data bases and networking.

Planning for information systems has a time horizon and a focus dimension. The time horizon dimension specifies the time range of the plan, whereas the focus dimension relates whether the primary concern is strategic, managerial, or operational.

2.2 BASIS FOR PLANNING IN SYSTEM ANALYSIS

System planning is the first phase in the system development life cycle. System planning is where an organization's total information needs are identified, analyzed, prioritized and arranged. Organization creates and assesses the original goals and expectation of a new system.

Importance of System Planning

The planning process provides the information top management needs to make effective decisions about how to allocate the resources in a way that will enable the organization to reach its objectives. Productivity is maximized and resources are not wasted on projects with little chance of success.

System Planning Process

The planning process includes the collection of all relevant system data, the modelling of the system using sophisticated software, the identification of key issues for consideration and solid recommendations designed to ensure proper and fiscally sound system development and operation.

The act, process, or profession of studying an activity (such as a procedure, a business, or a physiological function) typically by mathematical means in order to define its goals or purposes and to discover operations and procedures for accomplishing them most efficiently.

Tools and Techniques of System Analysis:

- **Grid Charts:** Grid charts are used to represent the relationship between two sets of factors in a tabular method.
- **System Flow Chart:** System flowcharts are a way of displaying how data flows in a system and how decisions are made to control events. To illustrate this, symbols are used. They are connected together to show what happens to data and where it goes. Note that system flow charts are very similar to data flow charts.
- **Decision Tree:** Decision tree is a tree like structure that represents the various conditions and the subsequent possible actions. It also shows the priority in which the conditions are to be tested or addressed. Each of its branches stands for any one of the

logical alternatives and because of the branch structure, it is known as a tree.

- **Simulation:** Simulation analysis is a model that is applied to analyze large projects and determine how target variables are affected based on changes in input variables. The model uses simulations to predict how the outcome of a decision would vary if we tweak a set of input variables in a given range.
- **Decision Tables:** Decision tables are used in various engineering fields to represent complex logical relationships. This testing is a very effective tool in testing the software and its requirements management. The output may be dependent on many input conditions and decision tables give a tabular view of various combinations of input conditions and these conditions are in the form of True(T) and False(F). Also, it provides a set of conditions and its corresponding actions required in the testing.

A requirement is vital feature of a new system which may include processing or capturing of data, controlling the activities of business, producing information and supporting the management. A system is made to solve a problem.

The process therefore has to start with recognition of the need. This step is called problem definition. In this phase the key question to be answered is "what is the problem that the system has to solve?"

This phase involves initial investigation and survey and should result in a clear statement of the scope and objectives of the system (See Fig. 1). It should give a clear idea of what is expected from the system in terms of performance criteria.

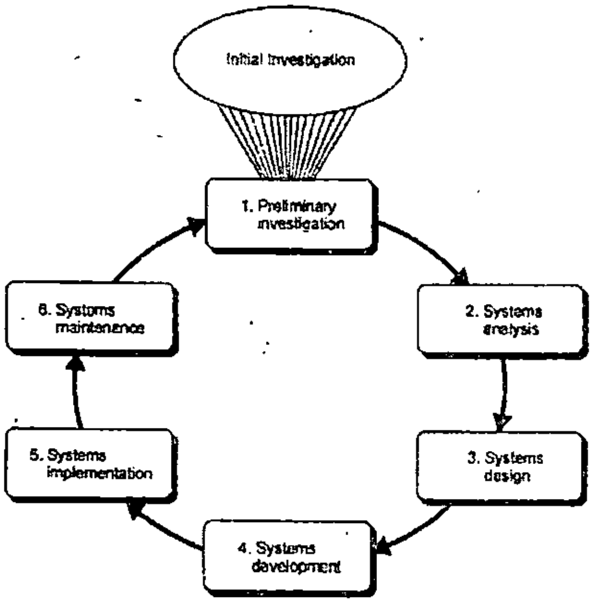


Fig. 1. The SDLC with preliminary investigation phase highlighted.

The identification of a need is the first step in the system development life cycle. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such requests, standard procedures must be established to deal with them. The initial investigation is one way of handling such request. The objective is to determine

Notes

whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system or existing one.

The identification of a need is the first step in the system development life cycle. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such requests, standard procedures must be established to deal with them. The initial investigation is one way of handling such request. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system or existing one.

2.3 PLANING AND CONTROL FOR SYSTEM SUCCESS

What can the analyst do to ensure the success of a system? First, a plan must be devised, detailing the procedure, some methodology, activities, resources, costs, and timetable for completing the system. Second, in larger projects, a project team must be formed of analysts, programmers, a system consultant, and user representatives. Shared knowledge, interaction, and the coordination realized through team effort can be extremely effective in contrast with individual analysts doing the same work.

Finally, the project should be divided into manageable modules to reflect the phases of system development - analysis, design, and implementation. Most of this work falls under project management and control. The main idea behind the system development life cycle is to formalize a means structured at three major levels for effective control of the project. At the lowest level, work assignments are broken down into small manageable tasks. A task is usually a well - defined, structured work unit that can be carried out by one individual. The task can be easily budgeted and scheduled and its quality measured. It can be easily completed independent of other tasks and other project team members. If rework is necessary, there is minimal loss or impact on other tasks, except where time is critical.

The second level at which work units are structured involves activities that have larger scope and are designed to produce substantial results. An activity is a group of logically related tasks that serve one phase of the system development life cycle. A phase, a third level of control, is a set of activities that bring the project to a ritical milestone. Milestones are steppingstones that make up the entire project.

In planning a project, the following steps should be taken:

1. Identify the activities in each phase and the tasks within each activity.
2. Calculate the budget for each phase and obtain agreement to proceed.
3. Review, record, and summarize progress on activities periodically.
4. Prepare a project progress report at the end of a reporting month

In summary, system development should not be regarded merely as some procedure that deals with hardware and software. The original assumptions upon which system specifications were based should be tested and re-evaluated with the user in mind.

Managing system projects includes the important responsibility of seeing to it that all features of the candidate system - technological, logical, and behavioural - are considered before implementation and maintenance.

Conclusion

System analysis and design are keyed to the system development life cycle(SDLC). The stages are project selection, feasibility, analysis, Design, implementation, and post implementation stages. The idea for the project is originates in the environment or from within the organization. Once the problem is verified an initial investigation is conducted to determines whether change is feasible. If the answer is yes, a feasibility study is conducted. Analysis is a detailed study of the various operation performed by a system. System design refer to the technical specifications that will be applied in implementing the candidate system. Implementation is concerned with details of the candidate system. After implementation, maintenance begins includes enhancements, modifications, or any changes from the original specifications.

To ensure the success of the system, careful and often extensive planning is required. The overall management process is crucial to the successful completion of system.

2.4 DIMENSIONS OF PLANNING

Long-term planning is something of a rarity amongst event organizers, and is therefore one of the key areas where our clients sees the most significant improvements upon using our services. Strategic planning stems from the vision, objectives, philosophy and policy formulation of the firm and extends its influence over longer time periods. Implementing a strategy entails making changes to current operations with future goals in mind.

An example would be using goal attainment to reach a zero CO2 emission goal for an annual event over a 10 year period. For every event, the objective is to reduce emissions by 10% towards the final goal. The firm is thereby given the opportunity to gradually adapt and do so cost-effectively over this period.

Other potential usages for strategic planning and goal setting are within product development. A strategy of this nature could focus on widening the range of events on offer in the firm's/destination's portfolio or expanding the size and contents of existing events.

To fully understand the new role of strategic quality planning, it is necessary to understand the dimensional nature of quality. Quality's five dimensions--experience, measurement, relationships, interconnectivity, and value sharing--provide a basis for understanding and defining quality.

These dimensions are repetitiveness (the extent to which the plan is used time after time); time ([the length of time covered by the plan); scope [the portion of total management which the plan is aimed at) and level [the levels of the organization which the plan is aimed at)

Likewise, what are three types of planning? There are three major types of planning, which include operational, tactical and strategic planning. A fourth type of planning, known as contingency planning, is an alternative course of action, which can be implemented if and when an original plan fails to produce the anticipated result.

Planning Process

The five process steps are:

Notes

- Set Objectives for the long run.
- Generate Alternative Strategies.
- Evaluate alternative strategies by comparison.
- Monitor strategies implementation and results.

Obtain a high level of commitment among the Stakeholders during each step of this process.

The planning process is the steps a company takes to develop budgets to guide its future activities. The documents developed may include strategic plans, tactical plans, operating plans, and project plans. The steps in the planning process are: Develop objectives. Develop tasks to meet those objectives.

2.5 INITIAL INVESTIGATION

The initial investigation is one way of handling such request. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system or existing one.

Systems Planning and the Initial Investigation - Systems This is a user's request to change, improve or enhance an existing system. The objective is to determine whether the request is valid or feasible The user request identifies the need for change and authorizes the initial investigation.

Secondly, what is preliminary investigation in system analysis and design? SYSTEM ANALYSIS Preliminary Investigation. Preliminary Investigation basically refers to the collection of information that guides the management of an organization to evaluate the merits and demerits of the project request and make an informed judgment about the feasibility of the proposed system.

One may also ask, what do you mean by initial investigation?

Initial Investigation is the first phase of SDLC and is known as. identification of need. This is a user's request to change, improve or. enhance an existing system.

Difference between Initial Investigation and Feasibility Study

The initial investigation is the primary step in the investigation stage of the project. A feasibility study is a research, usually done by technicians, that authenticates whether situations are right to execute a distinct project.

This is the first phase of SDLC and is known as identification of need.

- This is a user's request to change, improve or enhance an existing system.
- The objective is to determine whether the request is valid or feasible
- The user request identifies the need for change and authorizes the initial investigation.

The identification of a need is the first step in the system development life cycle. This is a user's request to change, improve, or enhance an existing system.

Because there is likely to be a stream of such requests, standard procedures must be established to deal with them. The initial investigation is one way of handling such request. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system or existing one.

The user's request form specifies the following:

1. User-assigned title of work requested
2. Nature of work requested
3. The date the request was submitted.
4. The date the job should be completed
5. Job objectives – purpose of job requested
6. Expected benefits to be derived from proposed change
7. Input/output description – quantity of inputs and outputs of proposed change
8. Requester's signature, title, department and phone number
9. Signature, title, department and phone number of person approving the request.

The user request identifies the need for change and authorizes the initial investigation. It may undergo several modifications before it becomes a written commitment. Once the request is approved, the following activities are carried out

- Background investigation
- Fact-finding and analysis
- Presentation of results – called project proposal.

2.6 PERFORMING INITIAL INVESTIGATION

The first step in the system development life cycle requires the identification of a need. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such requests, standard procedures must be established to deal with them. The initial investigation is one way of handling this. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system, or build a new one.

The first step in the system development life cycle requires the identification of a need. This is a user's request to change, improve, or enhance an existing system. Because there is likely to be a stream of such re.quests, standard procedu.tes must be established to deal with them. The initial investigation is one way of handling this. The objective is to determine whether the request is valid and feasible before a recommendation is reached to do nothing, improve or modify the existing system, or build a new one.

1. User-assigned title of work requested.
2. Nature of work requested (problem definition).
3. Date request to be submitted.
4. Date job should be completed.

5. Job objective(s)-purpose of job requested.
6. Expected benefits to be derived from proposed change.
7. Input/output description-quantity (number of copies or pages) and frequency (daily, weekly, etc.) of inputs and outputs of proposed change.
8. Requester's signature, title, department and phone number.
9.- Signature, title, department, and phone number of person approving the request.

The user request identifies the need for change and authorises the initial investigation. It may undergo several modifications before it becomes a written commitment. Once the request is approved, the following activities are carried out: background investigation, fact-finding and analysis, and presentation of results-called project proposal.

The proposal, when approved, initiates a detailed user-oriented specification of system petfotmance and analysis of the feasibility of the candidate system. A feasibility study focuses on identifying and evaluating alternative candidate systems with' a recommendation of the best sys.tem for the job. This chapter deals with the initial investigation. Chapter 4 discusses the feasibility study.

The user or the system analyst may identify the need for a candidate system or for enhancements in the existing system.

Often problems come into focus after a joint meeting between the user and the analyst. In either case, the user initiates an investigation by filling out a request form for information. The request provides for statements ofobjective and expected benefits.

The problem definition should clearly given:

- The objective
- The results which the user wants to achieve with the system. The problem must not be confused with the solution.

For example the following is not a problem:

We used a computerized database of customers."

This statement given no idea of what are the objectives of the system or what results a user wants. The problem in this case could have been any of the following:

We need to be able to generate a mailing list of all our old customers based on selection criteria like products purchased from us, city, income etc.

or

Our MD (Managing Director) needs to be able to claim, that we use computers at the next industry meeting to enhance out corporate.

or

We need to be able to effectively follow up pending payments as e have too many outstanding.

The problem should be separate from the symptoms and the causes and the solutions. An analyst may.start off with a statement like "the line in front of the teller is too long and find the problem which could be related to excessive processing time or a badly laid out building or a clerk who always comes in late.

The analyst needs to be alert and inquiring to define the actual problem and not get misled at this initial stage by possible solutions.

Stating the problems is the initial step of the project, Once the problem is defined, the project is initiated and the planning moves to the next step, where some background analysis is performed.

2.7 DATA AND FACT GATHERING TECHNIQUES

Fact finding is an important step for the analyst, based on which the analyst gains an understanding of the existing system and its problem, and of the requirement from the new system. There are a number of fact finding tools, which the analyst uses fur this; these are discussed in the following sub-section. The analyst chooses the tools he considers suitable and uses them to gather all the required in:fotmattoni which is then analyzed.

Study Basting Documents and Records

Many systems and organizations have some type of documents and records already available. Examples of such documents are:

- Input form
- Existing system user manuals, and other system documentation
- System review/audit
- Correspondence, for example, problem logs, request for enhancements
- Brochures
- Reports from the system
- Data file procedure manuals.

Study of available documentation is a fast and person-independent way of fact gathering. The analyst can study these document comfortably and use these to prepare further questions for the rest of the fact gathering exercise.

Interviews

Personal interviews are a direct method of obtaining information from people. By this the analyst learns about the existing system, problems and expectations.

During an interview the analyst and the person being interviewed are meeting face to face.This givsthe analyst a flexible opportunity to learn various pertinent facts. As the interviewer is facing the interviewee, he can observe the reaction of the interviewee and see how the person being asked a question is responding to it. This gives the interviewer an idea of the reliability of the information being gathered. It also gives the interviewer chance to ask more questions along with some promising line, which may not have been thought of earlier.

Notes

Questionnaires

A questionnaire seeks information from persons in a written form and in a prescribe format against a set of questions. Questionnaires are good for gathering specific information where the questions can be structured in advance and uniformly for a number of persons.

Questionnaires are useful as they are a quick means of gathering information and analyzing it together. They are particularly useful if the respondents are scattered geographically or there is no time to hold interviews.

Questions could be:

- Structured—here the respondent has to select from possible options and the range of answers is limited.
- Unstructured—asking the respondent's opining and letting the respondent answer freely. Such questions are open-ended.

Examples of structured questions are where the answer could be:

- Yes/no type
- Selected from specified multiple choice
- Selections on a ranking scale
- Selections of a rating
- Fill in the blank.

Structured questions need more analysis and may or may not give the type of information sought. Also, analyzing them may introduce analyst bias. They may however give better insights into the problems as they are open ended and exploratory. Often, they need follow up in the form of interviews.

Group Communication

When information is required from face to-face sessions, but there is not enough time to conduct personal interviews, meetings can be held. Here, since there are many persons present, more types ofideas can be discussed in a short time. Also, the comments of one person may prompt other persons to contribute facts they may have otherwise forgotten.

Conducting a group session is a skilled matter as there are problems like:

- The group may be dominated by some persons and others, who may otherwise have something to contribute, may be shy and not speak up.
- Problems with seniors may not be voiced because of seniors being present.
- The situation could lead to a verbal fight between persons which may need moderation.
- Internal politics of an organization may determine what is said and what. is left unsaid, thus resulting in a false picture.
- There may be situations where may persons talk simultaneously and are agitated and the comments they make may not be heard or get recorded.

Presentations

At times, the analyst may hold a presentation presenting his understanding of the system and problems etc. Such a presentation would typically involve showing slides and talking to a group of users whose response is sought. The persons attending the presentation have an opportunity to respond and confirm of affect the analyst's understanding.

Presentations are useful in situations where the users are passive or too busy to actively explain things. Here, an analyst may use study of existing records and questionnaires etc., to put together a presentation.

Fact Analysis

Data gathered by analyst has to be organized and evaluated to draw conclusions. Various means used to document and anlyze the data gathered include:

- Flow charts
- DFDs
- Decision tables
- Structure charts.

Analysis is done using techniques like data elements:

- Input-output
- Recurring data
- Reports usage.

Concluding the Initial Investigation

By this stage, the analyst has a thorough knowledge of tl)e system as part of the initial investigation. The initial investigation culminates with a system proposal or a project request;

Work on the initial investigation was initiated with a system request or an information system request which gave the 8)"stem objective, benefit expected, output descriptions, input descriptions. This document is modified as a result of the initial investigation, even the objectives may be modified and refined. The benefits and the input and output descriptions are expanded. The requirements for the feasibility study resources are also included.

The modified system request is presented by the analyst and is reviewed by the user. Review comments of the user are then incorporated to generate a final document.

Determining Project Benefits

An information system can provide many benefits to an organization. For example a new or renovated information system can automate monotonous jobs; and reduce errors; provide innovative services to customers and suppliers; and improve organizational efficiency, speed, flexibility, and. morale.

In general, the benefits can be viewed as being both tangible and intangible. Tangible benefits refer to items that can be measured in rupees and with certainty. Examples of tangible

Notes

benefits might include reduced personnel expenses, lower transaction costs, or higher profit margins. It is important to note that not all tangible benefits can be easily quantified. For example, a tangible benefit that allows a company to perform a task in 50 percent of the time may be difficult to quantity in terms of hard rupee savings. Most tangible benefits will fit within the following categories:

- Cost reduction and avoidance
- Error reduction
- Increased flexibility
- Increased speed of activity
- Improvement of management planning and conttol
- Opening new markets and increasing sales opportunities

Intangible benefits refer to items that cannotbe easily measured in rupees or with certainty. Intangible benefits, such as the improvement of employees morale, or they may have broader societal implications, such as the reduction of waste creation or resources consumption. Potential tangible benefits may have direct organizational benefits, such as the improvement of employees morale, or they may have broader societal implications, such as the reduction of waste creation or resources consumption.

Potential tangible benefits may have to be considered intangible during preliminary investigation because an analyst may not be able to quantify them in rupees or with certainty at the stage in the life cycle. During later stages, such intangibles can become tangible benefits as he/she better understands the ramifications of the system he/she is designing.

Determining Project Costs

Similar to benefits, an information system can have both tangible and intangible costs. Tangible costs refer to items that all analyst can easily measure in rupees and with certainty. From an IS development perspective, tangible costs include items such as hardware costs, labour costs, and operational costs including employee training and wilding renovations. Alternatively, intangible costs are those items that an analyst cannot easily measure in terms of rupees or with certainty. Intangible costs can include loss of customer goodwill, employee morale, or operational inefficiency.

Table 1 provides a summary of common costs associatea with the development and operation of an information system.

Predicting the costs associated with the development of an information system is an inexact science. IS researchers, however, have identified several guidelines for improving the cost-estimating process.

- Assign the initial estimating task to the final developers.
- Delay finalizing the initial estimate until the end of a thorough study.
- Anticipate and control user changes.
- Monitor the progress of the proposed project.
- Evaluate proposed project progress by using independent auditors.

- Use the estimate to evaluate project personnel.
- Study the cost estimate carefully before approving it.
- Rely on documented facts, standards, and simple arithmetic formulas rather than guessing, intuition, personal memory, and complex formulas, move as in other tables.
- Do not rely on cost-estimating software for an accurate estimate.

Table 1. Summary of common costs associated with the development and operation of an information system.

Types of Costs	*Examples*
Procurement	• Consulting costs • Equipment purchase or lease Equipment installation costs • Site preparation and modifications Capital costs • Management and staff time
Start-up	• Operating system software Communications equipment installation Start-up personnel • Personnel searches and hiring activities Disruption to the rest of the organization Management to direct start-up activity
Project-related	• Application software • Software modifications to fit locld systems Personnel, overhead, from in-house development Training users in application use • Collecting and analyzing data Preparing documentation Managing development
Operating	• System maintenance costs (hardware, software, and facilities) • Rental of space and equipment Asset depreciation • Management, operation, and planning personnel

Notes

2.8 NEEDS IDENTIFICATION

The success of a system depends largely on how accurately a problem is defined, thoroughly investigated and properly carried out through the choice of solution. Needs identification and analysis are concerned with what the user needs rather than what she wants. The analyst starts to think of the solution for the problem only after it has been identified, defined and evaluated. This helps the user and the analyst understand the real problem rather than its symptoms.

The user or the analyst may identify the need for a candidate system or enhancements in the existing system. The user initiates an investigation by filling out a request form for information. The request contains the objectives and expected benefits. (See figure - A sample user's request form).

Thus for the design of a successful system, a clear knowledge of the system is essential. This information is given out in needs identification.

Job Title:	Nature of job: (new or revision of existing one)		Request date	Date of completion
Job Objective:				
Expected Benefits:				
Output Specifications		Input Specifications		
Report title :		Document title :		
Quantity :		Quantity :		
Frequency :		Frequency :		
Remarks :		Remarks :		
Name of the Requester:	Signature:	Title:	Department:	Phone:
Approved By:	Signature:	Title:	Dept/Div:	Phone:
For MIS Dept only				
Job No:		Status:		
Acceptance authorized by:		Title:	Phone:	Remarks:

Identifying needs and requirements for a project takes time and effort. Don't be tempted to skip this step, assuming you know what the project demands. Conducting a comprehensive needs assessment as the project starts reduces the need for costly changes down the road. Depending on your company's organizational structure, you may generate these requirements in a top-down or bottom-up fashion. The top-down method comes from management, while the bottom-up method emerges organically from the organization's employees.

In either case or even a composite approach, the result directs the project outcome right from the beginning.

An organization can create a project using a top-down, bottom-up or a combined method. In the top-down method, upper management comes up with the ideas for a project. In the bottom-up method, employees and middle management create ideas for projects, and the combined method uses everyone in the organization to come up with ideas for a project.

An organization commonly assigns different teams to different parts of the project. By implementing a needs identification system, the organization helps to ensure the proper allocation of assets to different projects within the organization. Small-business owners should familiarize themselves with a basic list of project-management terms to better understand each project they start.

Identifying The Problems

Identifying potential problems before the start of a project can save the organization significant amounts of time and money. In order to look at the functional needs of the project (what the team needs to complete the project), the company must explain to the team the business needs of the project (the company's long-term strategic goals).

Problem analysis is one of the most critical stages of project planning because this stage helps to guide all subsequent analysis and decision-making. If the project does not advance past this stage with solutions that the organization can implement, the project should not go forward in its current form.

Reviewing Project Needs

The need for the project is identified after the organization makes observations about the project. Observations are often subjective and therefore someone with expertise about the proposed project should help to make observations.

A good observer can identify the needs of the project by answering key questions about the project. If the observations take into consideration the project itself and the outcome of the project, the observations should meet all of the needs of the project.

Gathering Basic Information

Observation and gathering information represent two processes. Observations highlight what is needed. On the other hand, gathering information highlights the processes needed to execute the proposed project. Both observations and the actual gathering of information should include comments from the group that ultimately will benefit from the completed project.

Objectives and Opportunities

Once the organization has analyzed the needs and identified the objectives, the organization needs to allocate funds to capitalize the project. By successfully identifying the needs, an organization can begin to allocate resources to pay for the project.

Additionally, a business needs to consider the potential future cash flow of the project. This allows the business to analyze potential cost savings to minimize costs and maximize the efficiency of the project.

Notes

2.9 DETERMINING THE USER'S INFORMATION REQUIREMENTS

User requirements form the essential building blocks for IT projects. Use these best practices to make sure you have a good set of requirements for your project. Managing user requirements is a specialized area you should address for any project that focuses specifically on identifying, gathering, communicating, and documenting client requirements for an IT system.

Once identified, the user requirements effectively lay the foundation for developers, testers, and implementers to begin determining the functionality, responsiveness, and interoperability required of that system. Unfortunately, many people consider gathering user requirements as a waste of time. However, the strategy is crucial to a project's success for developers and project managers to obtain accurate user requirements as well as increase the level of client and user involvement on a project.

Leading research indicates that poor requirements definition is a huge problem for many IT organizations. Unrealistic expectations that clients and users have regarding projects often arise because projects get started before the user requirements are determined. The biggest problems encountered on failed IT projects are:

- Misunderstood requirements and scope creep, which often cause an over-allocation in resources, additional cost or overdue deliverables.
- Incomplete requirements (data, reports), which result in incomplete information about the system.
- Unstable requirements or new requirements being added during the development phase.
- Ambiguity regarding the functionality and objectives of certain requirements.
- Conflicting goals by different teams (e.g., the users may want one thing and the business requires another approach).
- Too many "nice-to-have's" that wouldn't actually be used.

Companies do not always include the project manager during the first phases of a project (i.e., requirements gathering). Instead, many organizations simply hand off a "wish list" to the project or development manager at the development or deployment stage. But this strategy can cause major problems. Managers need to take a step back and review the user requirements thoroughly with the team.

A common result of poorly defined requirements

Consider this scenario: You want to develop a Web site selling high-tech gadgets. You proceed to tell the project manager and developers that you want a dynamic site with a large database, lots of graphics, a shopping cart, and a nice discussion forum. They begin developing your site based on these specs. Then you realize that you really wanted to have a scroll bar on your main index page and a full video archive for your consulting services, and you didn't really need that big a database after all.

Reluctantly, the developer removes some code and adds a few more scripts, digs out more images from a photo stock catalog, and downloads some additional software, all of which adds weeks to the overall timeline and increases the cost of labor and materials.

This situation is the rule and not the exception in many IT projects. Looking back, you can see that the initial requirements established for the site were incomplete, ambiguous, and not measurable. Additionally, the requirements changed midway through the project, which led to the project schedule being extended as well as adding additional costs to the project. Requirement errors can cost 10 percent more to correct than other errors. At the maintenance stage, it could cost up to 20 percent more.

Obtaining the user requirements

Given that the starting point of all requirements gathering is really a verbal interchange between the business analyst and client, good communication is crucial. Project managers must set the correct tone from the start of the project during the definition or exploratory phase. As a project manager, you must understand that user requirements must be captured correctly, and they have to be realistic and achievable.

One of the first things that must happen during this definition phase is the requirements-gathering meeting. This meeting usually includes representatives of all affected staff and could include the following members:

Approaches for Getting Information Regarding the Users Requirements

There are three general approaches for getting information regarding the users requirements. They are

- Asking
- Getting information from the existing information system
- Prototyping.

Asking

This strategy obtains information from users by simply asking them about the requirements. It assumes a stable system where users are well informed and can overcome biases in defining their problem. There are three key asking methods.

1. **Questions:** Questions may be open-ended or closed. An open-ended question allows the respondent to formulate a response. It is used when feelings or opinions are important. A closed question requests one answer from a specific set of responses. It is used when factual responses are known.

2. **Brainstorming:** Brainstorming is a technique used for generating new ideas and obtaining general information requirements. This method is appropriate for getting non conventional solutions to problems. A guided approach to brainstroming asks each participant to define ideal solutions and then select the best one. It works well for users who have sound system knowledge but have the difficulty of accepting new ideas.

3. **Group consensus:** This method asks participants for their expectations regarding specific variables. Each participant fills out a questionnaire. The results are summarized and given

Notes

to participants along with a follow-up questionnaire. Participants are invited to change their responses. The results are again summarized and given back to the participants. This debate by questionnaire continues until participants responses have converged enough. This method is advantageous than brainstorming because the participants are not subjected to psychological pressure.

Information from Existing Information System

There are two methods in extracting information from an already existing system

1. **Data Analysis approach:** Determining information from an existing application is called the data analysis approach. It simply asks the user what information is currently received and what other information is required. It depends on the user for getting accurate information. The analyst examines all reports, discusses each piece of information with the user, and determines unfulfilled information needs by interviewing the user.

   The analyst is primarily involved in improving the existing flow of data to the.user. The data analysis method is ideal for making structured decisions, although it requires that users articulate their information requirements. A major drawback is a lack of established rules for obtaining and validating information needs that are not linked to organizational objectives.

2. **Decision Analysis:** This method breaks down a problem into parts, which allows the user to focus separately on the critical issues. It also determines policy and organizational objectives relevant to complete each major decision. The analyst and the user then refine the decision process and the information requirements for a final statement of information requirements.

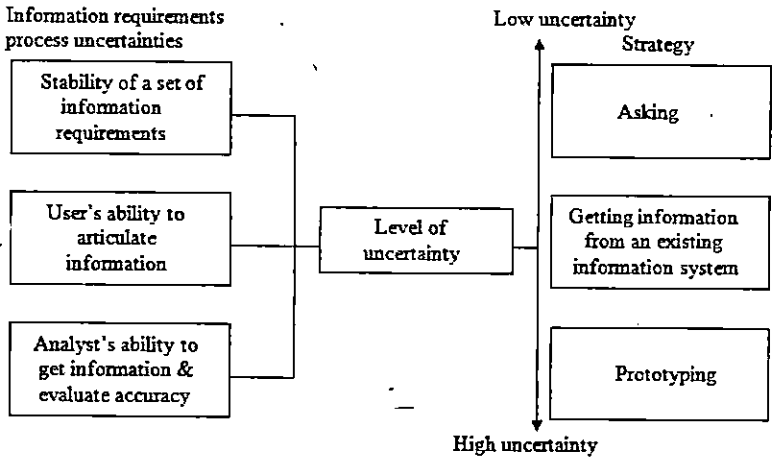
   In this method information needs are clearly linked to decision and organizational objectives. It is useful for unstructured decisions and information tailored to the user's decision-making style. The major drawback is that information requirements may change when the user is promoted or replaced

Protopying

The third strategy for determining user information requirements is used when the user cannot establish information needs accurately before the information system is built. The reason could be the lack of an existing model on which to decide requirements or a difficulty in visualizing candidate system.

In this case the user need to consider real life systems from which adjustments can be made. This iterative approach first set up the initial requirements and builds a system to meet these requirements.

As users gain experience, they request additional requirements or modifications and the process continues. Prototyping is suitable for environments where it is difficult to formulate a concrete model for defining information requirements. Prototyping strategy is appropriate for determining high uncertainty information requirement.


Information requirements process uncertainties
Stability of a set of information requirements
User's ability to articulate information
Analyst's ability to get information & evaluate accuracy
Level of uncertainty
Low uncertainty
Strategy
Asking
Getting information from an existing information system
Prototyping
High uncertainty

2.10 SYSTEM ANALYST

A systems analyst is an information technology (IT) professional who specializes in analyzing, designing and implementing information systems. Systems analysts assess the suitability of information systems in terms of their intended outcomes and liaise with end users, software vendors and programmers in order to achieve these outcomes.

A systems analyst is a person who uses analysis and design techniques to solve business problems using information technology. Systems analysts may serve as change agents who identify the organizational improvements needed, design systems to implement those changes, and train and motivate others to use the systems.

A systems analyst will often evaluate and modify code as well as review scripting. A system analyst is responsible for analyzing, designing and implementing systems to fulfil organizational needs. He/she plays a vital role in making operational the management information system.

Roles of System Analyst

- Identify, understand and plan for organizational and human impacts of planned systems, and ensure that new technical requirements are properly integrated with existing processes and skill sets.
- Plan a system flow from the ground up.
- Interact with internal users and customers to learn and document requirements that are then used to produce business required documents.
- Write technical requirements from a critical phase.
- Interact with software architect to understand software limitations.
- Help programmers during system development, e.g. provide use cases, flowcharts, UML and BPMN diagrams.

Notes

- Document requirements or contribute to user manuals.
- Whenever a development process is conducted, the system analyst is responsible for designing components and providing that information to the developer.

Systems analysts need to be able to understand humans' needs in interacting with technology, and they need enough computer experience to program, to understand the capabilities of computers, to glean information requirements from users, and to communicate what is needed to programmers.

System analysist is important because it provides an avenue for solutions in the system through the various tasks involved in doing the analysis. Through these various tasks, the overall quality of a system can be easily modified or improved and occurrences of errors can ultimately be reduced

Systems Analyst Job Responsibilities:

- Implements computer system requirements by defining and analyzing system problems; designing and testing standards and solutions.
- Defines application problem by conferring with clients; evaluating procedures and processes.
- Develops solution by preparing and evaluating alternative workflow solutions.
- Controls solution by establishing specifications and coordinating production with programmers.
- Validates results by testing programs.
- Ensures operation by training client personnel and providing support.
- Provides reference by writing documentation.
- Accomplishes information systems and organization mission by completing related results as needed.

Systems Analyst Qualifications / Skills:

- C
- COBOL
- Software design, documentation, testing, and maintenance
- Hardware requirements
- Teamwork
- General consulting skills
- Software architecture

Education, Experience, and Licensing Requirements:

- Bachelor's degree in computer science, mathematics, or engineering
- Experience in IT or database administration a plus
- Experience with Java GUI front-end development, SQL, Postgres, or equivalent database tools

- Experience with Agile software development using JIRA
- Experience with Zeek (formerly Bro)
- Experience in: Multiple OS platforms with strong emphasis on Linux (CentOS, Red Hat, Ubuntu), Mac OS X, and Windows systems
- Experience with OS-level scripting environment, such as Bourne shell
- Experience working in a research environment that relies extensively on Open Source solutions
- In-depth knowledge of the TCP / IP protocol suite; security architecture; securing and hardening Operating Systems; Networks; Databases; and Applications
- Knowledge of the Incident Response life-cycle, working independently to investigate and effectively respond to cyber security incidents
- Thorough understanding of the threat and attack landscape, attack vectors, vulnerabilities, and how they are leveraged by malicious actors
- Security certifications, Database Administrator certifications.

2.11 ANALYSIS

Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. A key question is, what must be done to solve the problem? One aspect of analysis is defining the boundaries of the system and determining whether or not a candidate system should consider other related systems. During analysis, data are collected on the available files, decision points, and transactions handled by the present system. Data flow diagrams interviews, on - site observations, and questionnaires are examples of the analysis tools. The interviews is a commonly used tool in analysis, it requires special skills and sensitivity to the subjects being interviewed. Bias in data collection and interpretation can be a problem. Training, experience, and common sense are required for collection of the information needed to do the analysis.

Once analysis is completed the analyst has a firm understanding of what is to be done. The next step is to decide how the problem might be solved. Thus, in systems, design we move from the logical to the physical aspects of the life cycle.

Design

The most creative and challenging phase of the system life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications (analogous to the engineer's blueprints) that will be applied in implementing the candidate system. It also includes the construction of programs and program testing. The key questions here is: How should the problem be solved? The major steps in design are:

The first step is to determine how the output is to be produced and in what format. Samples of the output (and input) are also presented.

Second, input data and master files (database) have to be designed to meet the requirements of the proposed output. The operational (processing) phases are handled through program construction and testing including a list of the programs needed to meet the system's objectives and complete documentation.

Finally, details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step toward implementation.

The final report prior to the implementation phase includes procedural flowcharts, record layouts, report layouts, and a workable plan for implementing the candidate system. Information on personnel, money, hardware, facilities, and their-estimated cost must also be available. At this point, projected costs must be close to actual costs of implementation.

In some firms, separate groups of programmers do the programming, whereas other firms employ analyst- programmers who do analysis and design as well as code programs. For this discussion, we assume that two separate persons carry out analysis and programming. There are certain functions, though, that the analyst must perform while programs are being written. Operating procedures must also be developed.

2.12 FEASIBILITY STUDY

During the first phase of the systems development life cycle, preliminary investigation, two primary activities are performed. The first project identification and selection, focuses on the activities during which the need for a new or enhanced system is re.cognized. This activity does not deal With a specific project but rather identifies the portfolio of projects to be undertaken by the organization.

Thus, project identification and selection is often thought of as a "preproject" step in the SDLC. This recognition of potential projects may come as part of a larger preliminary investigation process, information systems preliminary investigation, or from requests from managers and business units..

Regardless of how a project is identified and selected, the next step is to conduct a more detailed assessment during project identification and selection. This assessment does not focus on how the proposed system will operate but rather on understanding the scope of a proposed project and its feasibility of completion given the available resources. It is crucial that organizations understand whether resources should be devoted to a project; otherwise very expensive mistakes can be made.

Preliminary investigation is where projects are accepted for development, rejected, or redirected. This is also where a systems analyst plays a major .role in the systems development process. Numerous techniques for assessing project feasibility are described in this unit.

Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. A feasibility study is a test of a system proposal according to its workability. Impact on the organization, ability to meet user needs, and effective use of resources. It focuses on three major questions:

(i) What are the user's demonstrable needs and how does a candidate system meet them?

(ii) What resources are available for given candidate systems? Is the problem worth solving?

(iii) What is the likely impact of the candidate system on the organization? How well does it fit within the organization's master MIS plan?

Each of these questions must be answered carefully. They revolve around investigation and evaluation of the problem, identification and description of candidate systems, specification or performance and the cost of each system and final selection of the best system The objective of feasibility study is not to solve the problem but to acquire a sense of its scope. During the study the problem definition is crystallized and aspects of the problem to be included in the system are determined.

Consequently, costs and benefits are estimated with greater accuracy at this stage. The result of the feasibility study is a formal proposal. This is simply a report- a formal document detailing the nature and scope of the proposed solution. The proposal summarizes what is known and what is going to be done. It consists of the following:

1. *Statement of the problem* — a carefully worded statement of the problem that led to analysis.
2. *Summary of findings and recommendations*—a list of the major findingsand recommendations of the study. It is ideal for the user who requires quick access to the results of the analysis of the system under study. Conclusions are stated followed by a list of the recommendations and a justification for them.
3. *Details of findings*- an outline of the methods and procedures undertaken by the existing system followed by coverage of the objectives and procedures of the candidate system. Included are also discussions of output reports, file structures, and costs and benefits of the candidate system.
4. *Recommendations and conclusions*- specific recommendations regarding the candidate system including personnel assignments, costs, project schedules, and target dates.

After management reviews the proposal, it becomes a formal agreement that paves the way for actual design and implementations. This is a crucial decision point in the life cycle. Many project die here, whereas the more promising ones continue through implementations. Changes in the proposal are made in writing, depending on the complexity size, and cost of the project. It is simply common sense to verify changes before committing the project design.

Assessing Project Feasibility

All projects are. feasible given unlimited resources and infinite time. Unfortunately, most projects must be developed Within tight budgetary (i.e., financial) and time constraints. It means that assessing project feasibility is a required activity for all information systems projects and is a potentially large undertaking. It requites that a system analyst should evaluate a wide range of factors.

Typically, some of these factors will be more important than others for some projects and relatively unimportant for others. Although the specifies of a given project will dictate which factors are important, most feasibility factors are represented by the categories given below:

- Economic
- Technical
- Operational

- Schedule
- Legal and contractual
- Political

Together, the culmination of these feasibility analyses .forms the business case that justifies the expenditure of resources on the project. .Let us examine the various feasibility issues.

Assessing Economic Feasibility

The purpose of assessing econondc feasibility isto identify the financial benefits and costs associated with the development project. Economic feasibility is often referred to as cost-benefit analysis. During preliminary investigation, it will be impossible to precisely define all benefits and costs related to a particular project.

Yet, it is important that the analyst should spend adequate time identifying and quantifying these items or it will be impossible to conduct an adequate economic analysis and make meaningful comparisons between rival projects. Here, we will describe typical benefits and costs resulting from the development of an information system. Useful worksheets can also be provided for recording costs avd benefits.

Additionally, several common techniques for malting cost benefit calculations are presented. These workshcets and techniques are used after each SDLC phase as the project is reviewed in order to decide whether to continue, redirect, or kill (abandon) a project.

Testing Project Feasibility

Preliminary investigations examine project feasibility, the likelihood the system will be useful to the organization. Three tests of feasibility-all equally important are studied: operational, technical and financial.

Operational Feasibility

Proposed projects are beneficial only if they can be turned into information systems that will meet the organization's operating requirements. Simply stated, this test of feasibility ,asks if the system will work when it is developed and installed. Are there major barriers to implementation? Here are questions that will help test the operational feasibility of project:

- Is there sufficient support for the project from management? From users? If the current system is well liked and used to the extent that persons will not be able to see reasons for a change, there may be resistance.
- Are current business methods acceptable to the users? If they are not, users may welcome a change that will bring about a more operational and useful system.
- Have the users been involved in the planning and development of the project?
- Early involvement reduces the chances of resistance to the system and change in general and increases the likelihood of successful projects.
- Will the proposed system cause harm? Will it produce poorer result in any respect or area? Will loss of control result in any area? Will accessibility of information be lost? Will individual performance be poorer after implementation than before?

- Will customers be affected in an undesirable way? Will the system slow performance in any areas?
- Issues that appear to be relatively minor in the beginning have ways of growing into major problems after implementation. Therefore, all operational aspects must be considered carefully.

Technical Feasibility

The technical issues usually raised during the feasibility stage of the investigation include these:

1. Does the necessary technology exist to do what is suggested (and can it be acquired)?
2. Does the proposed equipment have the technical capacity to hold the data required to use the new system?
3. Will the proposed system provide adequate responses to inquiries, regardless of the number or location of users?
4. Can the system be expanded if developed?
5. Are there technical guarantees of accuracy, reliability, ease of access, and data security?

For example, if the proposal includes a printer that prints at the rate of 15,000 lines per minute, a brief search shows that this specification is technically feasible. (Whether it should be included in the configuration is an economic decision.) On the other hand, if a user is requesting voice input to write, read, and change stored data, the proposal may not be technically feasible.

Handling Infeasible Projects

Not all projects submitted for evaluation and review are judged acceptable. Requests that fail to pass feasibility tests are not pursued further, unless they are reworked and resubmitted as new proposals. In some cases, only part of a project is actually unworkable, and the selection committee may decide to combine the workable part of the project with another feasible proposal.

In still other cases, preliminary investigations produce enough new information to suggest that improvements in management and supervision, not the development of information systems, are the actual solutions to reported problems.

2.13 FEASIBILITY CONSIDERATIONS

Three key considerations are involved in the feasibility analysis: economic, technical and behavioral. Let's briefly review each consideration and how it relates to the systems effort.

Economic Feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of a candidate system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare

Notes

them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. Otherwise, further justification or alterations in the proposed system will have to be made if it is to have a chance of being approved.

This is an ongoing effort that improves in accuracy at each phase of the system life cycle.

A system that can be developed technically and that will be used installed must still be a good investment for the organization. Financial benefits must equal or exceed the costs. The financial and economic questions raised by analysts during he preliminary investigation are for the purpose of estimating the following:

1. The cost to conduct a full systems investigation
2. The cost of hardware and software for the class of application being considered.
3. The cost nothing changes (i.e., the proposed system is not developed

To be judged feasible, a project proposal must passed all these tests. Otherwise, it, is not a feasible project. For example, a personnel record feasible if the necessary technology does not exit. A medical system that can be developed at reasonable costs but that nurses will avoid using cannot be judged operationally feasible.

Technical Feasibility

Technical feasibility centers around the existing computer system (hardware, software, etc.) and to what extent it can support the proposed addition. For example, if the current computer is operating at 80 percent capacity—an arbitrary ceiling- then running another application could overload the system or require additional hardware. This involves financial considerations to accommodate technical enhancements. If the budget is a serious constraint, then the project is judged not feasible.

Behavioral Feasibility

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made of how strong a reaction the user staff is likely to have toward the development of a computerized system. It is common knowledge that computer installations have something to do with turnover, transfers, retraining and changes in employee job status. Therefore, it is understandable that the introduction of a candidate system requires special effort to educate, sell and train the staff on new ways of conducting business.

In safe deposit example, three employees are more than 50 years old and have been with the bank over 14 years, four of which have been in safe deposit. The remaining two employees are in their early thirties. They joined safe deposit about two years before the study. Based on data gathered from extensive interviews, the younger employees want the programmable aspects of safe deposit (essentially billing) put on a computer.

Two of the three older employees have voiced resistance to the idea. Their view is that billing is no problem. The main emphasis is customer service - personal contact with customers. The decision in this case was to go ahead and pursue the project.

2.14 STEPS IN FEASIBILITY ANALYSIS

Feasibility analysis involves eight steps:

1. From a project team and appoint a project leader.
2. Prepare system flowcharts.
3. Enumerate potential candidate systems.
4. Describe and identify characteristics of candidate systems.
5. Determine and evaluate performance and cost effectiveness of each candidate system.
6. Weight system performance and cost data.
7. Select the best candidate system.
8. Prepare and report final project directive to management

1. Form a project Team and Appoint a Project Leader

The concept behind a project team is that future system users should be involved in its design and implementation. Their knowledge and experience in the operations area are essential to the success of the system. For small projects, the analyst and an assistant usually suffice; however, more complex studies require a project team. The team consists of analysts and user staff - enough collective expertise to devise a solution to the problem. In many cases, an outside consultant and an information specialist join the team until the job is completed.

Projects are planned to occupy a specific time period, ranging from several weeks to months. The senior systems analyst is generally appointed as project leader. He/she is usually the most experienced analyst in the team. The appointment is temporary, lasting as long as the project.

Regular meetings take place to keep up the momentum and accomplish the mission – selection of the best candidate system. A record is kept of the progress made in each meeting. Regarding the safe deposit case, since the whole user area consists of five employees, the analyst handled most of the work.

2. Prepare System Flowcharts

The next step in the feasibility study is to prepare generalized system flowcharts for the system. Information – oriented charts and data flow diagrams prepared in the initial investigation are also reviewed at this time. The charts bring up the importance of inputs, outputs and data flow among key points in the existing system. All other flowcharts needed for detailed evaluation are completed at this point.

3. Enumerate Potential Candidate Systems

This step identifies the candidate systems that are capable of producing the outputs included in the generalized flowcharts. This requires a transformation from logical to physical system models. Another aspect of this step is consideration of the hardware that can handle the total system requirements. In the safe deposit case, it was found that virtually any microcomputer

Notes

system with more than 128k -byte memory an dual disk drive will do the job. It was also learned that a microcomputer can be designed to interface with the bank's mainframe. In this design, actual processing is handled by the microcomputer, whereas information such as payments and credits are transmitted to the main computer files for proper adjustment through the customer's checking account. The question here is: which microcomputer (IBM, Apple, Digital etc.) should be selected?

This is taken up in step 6 of the study.

An important aspect of hardware is processing and main memory. There are a large number of computers with differing processing sizes, main memory capabilities and software support. The project team may contact vendors for information on the processing capabilities of the system available.

4. Describe and Identify Characteristics of Candidate System

From the candidate systems considered, the team begins a preliminary evaluation in an attempt to reduce them to a manageable number. Technical knowledge and expertise in the hardware / software area are critical for determining what each candidate system can and cannot do. In the safe deposit example, a search for the available microcomputers and safe deposit billing packages revealed the information summarized.

These packages were the result of a preliminary evaluation of more than 15 other packages – all purporting to meet the requirements, of the safe deposit billing system. When the number is reduced to three key packages, the next step is to describe in some detail the characteristics of each package. For example, the first candidate system runs on an IBM PC with a minimum of 128K-bytes of memory.

The software is written in Oracle, a relatively new language. In case of enhancements, change has to be made through the software house, since the source code is not available to the user. The first package was installed in January 2002. More than 200 packages have been installed to date.

The next two candidate systems are similarly described. The information along with additional data available through the vendor highlights the positive and negative features of each system. The constraints unique to each system are also specified. For example, in the IBM PC package, the lack of an available source code means that the user has to secure a maintenance contract that costs 18 percent of the price of the package per year. In contrast the HP 100 package is less expensive and offers a source code to the user. A maintenance contract (optional) is available at 18 percent of the price of the package.

5. Determine and Evaluate Performance and Cost Effectiveness of Each Candidate System

Each candidate system's performance is evaluated against the system performance requirements set prior to the feasibility study. Whatever the criteria, there has to be as close a match as practicable, although trade-offs are often necessary to select the best system. In the safe deposit case, the criteria chosen in advance were accuracy, growth potential, response time less than five seconds, expandable main and auxiliary storage, and user-friendly software.

Often these characteristics do not lend themselves to quantitative measures. They are usually evaluated in qualitative terms (excellent, good, etc.) based on the subjective judgement of the project team. The cost encompasses both designing and installing the system. It includes user training, updating the physical facilities and documenting.

System performance criteria are evaluated against the cost of each system to determine which system is likely to be the most cost effective and also meets the performance requirements. The safe deposit problem is easy. The analyst can plot performance criteria and costs for each system to determine how each fares.

Costs are more easily determined when the benefits of the system are tangible and measurable. An additional factor to consider is the cost of the study design and development. The cost estimate of each phase of the safe deposit project was determined for the candidate system (IBM PC). In many respects, the cost of the study phase is a "sunk cost" (fixed cost). Including it in the project cost estimate is optional.

6. Weight System Performance and Cost Data

In some cases, the performance and cost data for each candidate system show which system is the best choice? This outcome terminates the feasibility study. Many times, however, the situation is not so clear - cut. The performance / cost evaluation matrix at times does not clearly identify the best system, so the next step is to weight the importance of each criterion by applying a rating figure. Then the candidate system with the highest total score is selected.

The procedure for weighting candidate systems is simple: -

1. Assign a weighting after factor to each evaluation criterion based on the criterion' effect on the success of the system. For example, if the usability criterion is twice as important as the accuracy factor, usability is assigned weight 4 and accuracy is assigned weight.
2. Assign a quantitative rating to each criterion's qualitative rating. For example, ratings (poor, fair, good, very good, excellent) may be assigned respective values (1,2,3,4,5).
3. Multiply the weight assigned to each category by the relative rating to determine the score.
4. Sum the score column for each candidate system.

Thus, the weighted candidate evaluation matrix is prepared using these steps, which in it self helps in the next step.

7. Select the Best Candidate System

The system with highest total score is judged the best system. This assumes the weighting factors are fair and the rating of each evaluation criterion is accurate. The criterion of growth potential is generally given the maximum weight, thus the greatest effect on the total score. Additionally, system development and user training are also given high weights.

Most feasibility studies select from more candidate systems than we have mentioned in our example. The criteria chosen and the constraints are also more complex. In any case, management should not make the selection without having the experience to do so. Management cooperation and comments, however, are encouraged.

Notes

2.15 FEASIBILITY REPORT

The culmination of the feasibility study is a feasibility report directed to management; it evaluates the impact of the proposed changes on the area(s) in question. The report is a formal document for management use, brief enough and sufficiently non-technical to be understandable, yet detailed enough to provide the basis for system design.

There is no standard format for preparing feasibility reports. Analysts usually decide on a format that suits the particular user and system. Most reports, however, begin with a summary of findings and recommendations, followed by document details. Starting with summary information highlights the essence of the report, giving management the option of reviewing the details later. The report contains the following sections:

1. Cover letter formally presents the report and briefly indicates to management the nature, general findings and recommendations to be considered.
2. Table of content specifies the location of the various parts of the report. Management quickly refers to the sections that concern them.
3. Overview is a narrative explanation of the purpose scope of the project, the reason for undertaking the feasibility study and the department(s) involved or affected by the candidate system. Also included are the names of the persons who conducted the study, when it began, and other information that explains the circumstance surrounding the study.
4. Detailed findings outline the methods used in the present system. The system's effectiveness and efficiency as well as operating costs are emphasized. The section also provides a description of the objectives and general procedures of the candidate system. A discussion of output reports, costs and benefits gives management a feel for the pros and cons of the candidate system.
5. Economic justification details point-by-point cost comparisons and preliminary cost estimates for the development and operation of the candidate system. A return on investment (ROI) analysis of the project is also included.
6. Recommendations and conclusions suggest to management the most beneficial and cost-effective system. They are written only as a recommendation, not a command. Following the recommendations, any conclusions from the study may, be included.
7. Appendixes document all memos and data compiled during the investigation. They are placed at the end of the report for reference. Disapproval of the feasibility report is rare if it has been conducted properly. When a feasibility team has maintained good rapport with the user and his/ her staff it makes the recommendations easier to approve. Technically, the report is only a recommendation, but it is an authoritative one. Management has the final say. Its approval is required before system design is initiated.

Oral Presentation

The feasibility report is a good written presentation documenting the activities involving the candidate system. The pivotal step, however, is selling the proposed change. Invariably the project leader or analyst is expected to give an oral presentation to the end user. Although it is not as polished as the written report, the oral presentation has several important objectives. The most critical requirements for the analyst who gives the oral presentation are:

(1) communication skills and knowledge about the candidate system that can be translated into language understandable to the user and

(2) the ability to answer questions, clarify issues, maintain credibility and pick up on any new ideas or suggestions.

The substance and form of the presentation depend largely on the purposes sought. The presentation may aim at informing, confirming, or persuading.

1. **Informing.** This simply means communicating the decisions already reached on system recommendations and the resulting action plans to those who will participate in the implementation. No detailed findings or conclusions are included.
2. **Confirming.** A presentation with this purpose verifies facts and recommendations already discussed and agreed upon. Unlike the persuading approach, no supportive evidence is presented to sell the proposed change, nor is there elaborate reasoning behind recommendations and conclusions. Although the presentation is not detailed, it should be complete. Confirming is itself part of the process of securing approval. It should reaffirm the benefits of the candidate system and provide a clear statement of results to be achieved.
3. **Persuading.** This is a presentation pitched toward selling ideas- attempts to convince executives to take action on recommendations for implementing a candidate system.

Regardless of the purpose sought, the effectiveness of the oral presentation depends on how successful the project team has been in gaining the confidence of frontline personnel during the initial investigation. How the recommendations are presented also has an impact. Here are some pointers on how to give an oral presentation:

1. Rehearse and test your ideas before the presentation. Show that you are in command. Appear relaxed.
2. Final recommendations are more easily accepted if they are presented as ideas for discussion, even though they seem to be settled and final.
3. The presentation should be brief, factual and interesting Clarity and persuasiveness are critical. Skill is needed to generate enthusiasm and interest throughout the presentation.
4. Use good organization. Distribute relevant material to the user and other parties in advance.
5. Visual aids (graphs, charts) are effective if they are simple, meaningful and imaginative. An effective graph should teach or tell what is to be communicated.
6. Most important, present the report in an appropriate physical environment where the acoustics, seating pattern, visual aid technology and refreshments are available.

The most important element to consider is the length of the presentation. The duration often depends on the complexity of the project, the interest of the user group and the competence of the project team.

Notes

A study that has company wide applications and took months to complete would require hours or longer to present. The user group that was involved at the outset would likely permit a lengthy presentation, although familiarity with the project often dictates a brief presentation.

Unfortunately, many oral presentations tend to be a rehash of the written document with little flare or excitement. Also, when the analyst or the project leader has a good reputation and success record from previous projects, the end user may request only a brief presentation.

Conclusion

A feasibility study is conducted to select the best system that meets performance requirements. A system required performance is defined by statement of constraints, the identification of specific system objectives, and a description of outputs. The analyst is ready to evaluate the feasibility of the candidate systems to produce these outputs. Three key considerations are involved in feasibility analysis are economic, technical, and behavioural feasibility. Feasibility analysis involves eight steps:

1. From a project team and appoint a project leader.
2. Prepare system flowcharts.
3. Enumerate potential candidate systems.
4. Describe and identify characteristics of candidate systems.
5. Determine and evaluate performance and cost effectiveness of each candidate system.
6. Weight system performance and cost data.
7. Select the best candidate system.
8. Prepare and report final project directive to management.

SUMMARY

- Planning information systems has become increasingly important because information is a vital reimurce and company asset. more and 'more funds are committed to information systems, and. system development is a serious business for computers that incorporate data 'bases and networking.
- Planning for information systems has a time horizon and a focus dimension. The time horizon dimension specifies the time range of the plan, whereas the focus dimension relates whether the primary concern is strategic, managerial, or operational.
- The initial investigation has the objective of determining the validity of the use.r's request for a candidate system and whether a feasibility study should be conducted. The objectives

of the problem posed by the user must be understood within the framework of the organization's MIS plan.

- Determining user requirements is not easy. System requirements change, the articulation of requirements is difficult, and heavy user involvement and motivation are uncertain. Problems with the user/analyst interface add further difficulties to the procedure.
- There are three strategies for eliciting information regarding the user's requirements: asking questions, obtaining information from the present system, and prototyping. The asking strutegy assumes a stable system where the user is well informed about information requirements. In CQntrast, the prototyping strategy is appropriate for high-uncertainty information requirements determination.
- Fact-finding is the first step in the initial investigation. It includes the review of written documents, on-site observations, interviews, and questionairs. The next step is fact analysis, which evaluates the elements related to the inputs and outputs of a given system. Data flow diagrams and other charts are prepared during this stage.
- Personal interriews are a direct method of obtaining information from people. Questionnaires are useful as they are a quick means of gathering information and analyzing it together.
- The data flow diagram CDFD) shows the flow of data, the proes. and the areas where they are stored. It is a commonly used structured tool for displaying the logical aspects of the system under study.Decision tables are used as a supplement when complex decision logic cannot be represented clearly in a DFD.
- The outcome of the initial investigation is to deterinine whether an alternative system is feasible. The proposal details the findings of .the investigation. Approval of the document initiates a feasibility study which leads to the selection of the best candidate system.

abc KEY WORDS

- **Lifelong Learning:** ongoing, voluntary, and self-motivated pursuit of knowledge for either personal or professional reasons client interface of a class.
- **Commonalities:** The set of features or properties of a component (or system) that are the same, or common, between systems
- **Compatibility:** The ease of combining software elements with others. The ability of two or more systems or components to perform their required functions while sharing the same hardware or software environment. The ability of two or more systems or components to exchange information.
- **Design:** The phase in the software life-cycle that emphasises a logical solution, i.e. how the system fulfills the requirements. During object-oriented design, there is an emphasis on defining logical software objects that will ultimately be implemented in an object-oriented programming language. In this view, the design serves as a high level description of the source code, describing its key features and giving a blueprint

Notes

of how the code is organised. The process of defining the architecture, components, interfaces, and other characteristics of a system or component.

- **Feasibility:** It is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software
- **Implementation:** The phase in the software life-cycle where the actual software is implemented. The result of this phase consists of source code, together with documentation to make the code more readable.
- **Maintainability:** The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment.

REVIEW QUESTIONS

1. What is the difference between analysis and design. Explain.
2. How would an analysis determine the users' needs for a system. Explain.
3. Distinguish between initial investigation and feasibility study. In what way are they related.
4. How does system design simplify implementation
5. What planning dimensions determine information system development? Elaborate.
6. Why is it difficult to determine user requirements.
7. What is the purpose of preliminary investigation.
8. How to conduct the feasibility study

FURTHER READINGS

1. Software Engineering, Bharat Bhushan Agarwal, Sumit Prakash Tayal, Firewall Media.
2. Softwaree Engineering, D. Sunder, University Science Press
3. Kaniclides, A., C. Kimble.(1995) A Development Framework for Executive Information Systems. Proceedings of GRONICS'95, Groningen, The Netherlands, February.
4. Kimble, C., (1997) Assessing the Relative Importance of Factors Affecting Information Systems Success, University of York Technical Report Series, YCS 283, Department of Computer Science, York, UK.

Unit 3

Tools of Structured Analysis

Notes

Structure

3.0 LEARNING OBJECTIVES

After reading this chapter students will be able to:

- know the tools of structured analysis
- understand the data flow diagram
- disucuss the entity relationship diagrams
- disucuss the data dictionary and process modelling
- explain the structured english, and decision tree and tables
- define object oriented analysis and design.

Notes

3.1 INTRODUCTION

Structured analysis is a set of techniques and graphical tools that allow the analyst to develop a new kind of system specifications that are easily understandable to the user. Analysts work primarily with their wits, pencil, and paper. Most of them have no tools. The traditional approach focuses on cost/benefit and feasibility analysis, project management, hardware and software selection and personnel considerations. In contrast, structured analysis considers new goals and structured tools for analysis. The new goals specify the following:

1. Use graphics wherever possible to help communicate better with the user.
2. Differentiate between logical and physical systems.
3. Build a logical system model to familiarize the user with system characteristics and interrelationships before implementation.

The structured tools focus on the listed earlier- essentially the date flow diagram data dictionary, structured English, decision trees, and decision tables. The objective is to build a new document, called system specifications. This document provides the basis for design and implementation. The system development life cycle with structured analysis. The primary steps are:

- Process 2.1: Study affected user areas, resulting in a physical DFD. The logical equivalent of the present system results in a logical DFD.
- Process 2.2: Remove the physical checkpoints and replace them with a logical equivalent, resulting in the logical DFD.
- Process 2.3: Model new logical system. So far no consideration is given to modifying methods called for in the feasibility report. This step incorporates the changes the begins to describe the candidate system. It is essentially a paper model system to be installed.
- Process 2.4: Establish man/ machine interface. This process modifies the logical DFD for the candidate system and considers the hardware needed to implement the system. The combination results in the physical DFD of the candidate system.
- Process 2.5 and 2.6: Quantify costs and benefits and select hardware. The purpose of this step is to cost- justify the system, leading to the selection of hardware for the candidate system. All that is left after this step is writing the structured specification.

The structured specification consists of the DFDs that show the major decomposition of system functions and their interfaces, the data dictionary documenting all interface flows and data stores on the DFDs and documentation of the intervals of DFDs in a rigorous manner through structured English, decision trees, and decision tables.

In conclusion, structured analysis has the following attributes:

1. It is graphic. The DFD for example, presents a picture of what is being specified and is a conceptually easy-to - understand presentation of the application.
2. The process is partitioned so that we have a clear picture of the progression from general to specific in the system flow.

3. It is logical rather than physical. The elements of system do not depend on vendor or hardware. They specify in a precise, concise, and highly readable manner the working of the system and how it hangs together.
4. It calls for a rigorous study of the user area, a commitment that is often taken lightly in the traditional approach to systems analysis.
5. Certain tasks that are normally carried out late in the system development life cycle are moved to the analysis phase. For example, user procedures are documented during rather than later in implementation.

3.2 THE DATA FLOW DIAGRAM (DFD)

The first step is to draw a data flow diagram (DFD). The DFD was first developed by Larry Constantine as a way of expressing system requirements in a graphical from; this led to a modular design.

A DFD also known as a "bubble chart," has the purpose of clarifying system requirements and identifying major transformations that will be come programs in system design. So it is the starting point of the design phase that functionally decompose the requirements specifications down to the lowest level of detail.

A DFD consists of a series of ' ·bbles joined by lines. The bubbles represent data transformations and the lines represeı.. data flows in the system. The system takes orders from the customer (bookstore, library, etc.), checks them against an index (file) listing the books available, verifies customer credit through a credit information file, and authorizes, shipment with an invoice.

DFD Symbols

In the DFD, there are four symbols A square defines a source (originator) or destination of system data.

1. An arrow identifies data flow- data in motion. It is a pipeline through which information flows.
2. A circle or a "bubble" (some people use an oval bubble) represents a process that transforms incoming data flow(s) into outgoing data flow(s).
3. An open rectangle is a data store- data at rest, or a temporary repository of data.

   Note that a DFD describes what data flow (logical) rather than how they are processed so it does not depend on hardware, software, data structure, or the organization. The key question that we are trying to answer is: What major transformations must occur for input to be conectly transformed into output? Elaborate on the logical functions of the system. first, incoming orders are checked for correct book titles, author's names and other information and their batched with other book orders from the same bookstore to determine how many copies can be shipped through the warehouse. Also, the credit status of each bookstore is checked before shipment is authorized. Each shipment has a shipping notice detailing the kind and number of books shipped. This is compared to the original order received (by mail or phone) to ascertain its accuracy. The details of the order are normally available in a special file or a data store, called "bookstore orders."

Following order verification and credit check, a clerk batches the order by assembling all the book titles ordered by the bookstore. The batched order is sent to the warehouse with authorization to pack and ship the books to the customer.

Further expansion of the DFD focuses on the steps taken in billing the bookstore. As you can tell by now, each process summarizes a lot of information and can be exploded into several lower-level detailed DFDs. This is often necessary to make sure that a complete documentation of the data flow is available for further reference.

Constructing a DFD

Several rules of thumb are used in drawing DFDs.

1. Process should be named and numbered for easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from the source (upper left corner) to the destination (lower right corner), although they may flow back to a source. One way to indicate this is to draw a long flow line back to the source. An alternative way is to repeat the source symbol as a destination. Since it is used more than one in the DFD, it is marked with a short diagonal in the lower right corner.
3. When a process is exploded into lower- level details, they are numbered.
4. The names of data stores, sources and destinations are written in capital letters.

Process and data flow names have the first letter of each word capitalized. How detailed should a DFD be? As mentioned earlier, the DFD is designed to aid communication. If it contains dozens of processes and data stores, it gets too unwieldy.

The rule of thumb is to explode the DFD to a functional level, so that the next sub level does not exceeded 10 processes. Beyond that, it is best to take each function separately and expand it to show the explosion of the single process.

If a user wants to know what happens within a given process, then the detailed explosion of that process may be shown.

A DFD typically shows the minimum contents of data stores. Each data store should contain all the data elements that flow in and out. Questionnaires can be used to provide information for a first cut. All discrepancies, missing interfaces, redundancies, and the like are then accounted for- often through interviews.

The DFD methodology is quite effective, especially when the required design is unclear and the user and the analyst need a notational language for communication. The DFD is easy to understand after a brief orientation.

The main problem however is the large number of iterations that often are required to arrive at the most accurate and complete solution.

Data Dictionary

In our data flow diagrams, we give names to data flows, processes and data stores. Although the names are descriptive of the data, they do not give details. So following the DFD our interest is to build some structured pace to keep details of the contents of data flows, processes and data store.

A data dictionary is a structured repository of data. It is a set of rigorous definitions of all DFD data elements and data structure.

A data dictionary has many advantages. The most obvious is documentation: it is a valuable reference in any organization. Another advantage is improving analyst/ user communication by establishing consistent definitions of various elements, terms and procedures. During implementation, it serves as a common base against which programmers who are working on the system compare their data descriptions.

Also control information maintained for each data element is cross- referenced in the data dictionary. For example, programs that use a given data element are cross- referenced in a data dictionary, which makes it easy to identify them and make any necessary changes.

Finally a data dictionary is an important step in building a database. Most data base management systems have a data dictionary as a standard feature. Data have been described in different ways. For example, in tape and disk processing, IBM called a file data set. In data base technology, the term file took on a different meaning IBM's information Management

System's (IMS) manual defines data as fields divided into segments, which, in turn, are combined into databases. The Conference on Data System Languages (CODASYL) defines data as data items combined into aggregates, which, in turn are combined into records. A group of related records is referred to as a set. If we choose words that represent the general thinking of common vocabulary. There are three classes of items to be defined:

1. Data element: The smallest unit of data that provides for no further decomposition. For example, " data" consists of day, months and year. They hand together for all practical purposes.
2. Data structure: a group of data elements handled as a unit. For example, " phone" is a data structure consisting of four data elements: Area code- exchange – number –extension- for example, 804-924-3423-236. "BOOK DETAILS" is a data structure consisting of the data elements author name, title, ISBN (International Standard Book Number), LOCN (Library of Congress Number), publisher's name and quantity.
3. Data flows and data stores. As defined earlier, data flows are data structures in motion, whereas data stores are data structures at rest. A data store is a location where data structures are temporarily located.

Describing Data Elements

The description of a data element should include the name, description and an alias (synonym). For example:

AUTHOR NAME	-first	WHISKEY	- name
	- middle		- distiller

Notes

- last
- vintage
- alias

The description should be a summary of the data element. It may include an example. We may also want to include whether or not the data elements(s) has:

1. A different name. For example a PURCHASE ORDER may exist as PUR.ORDER, PUCHASE ORD., or P.O. We want to record all these in the data dictionary and include them under the PUCHASE ORDER definition and separately with entires of their own. One example is "P.O. alias of (or see also) PUCHASE ORDER." Then we look up PUCHASE ORDER to find the details. It is an index.
2. Usage characteristics, such as a range of values or the frequency of use or both. A value is a code that represents a meaning. Here we have two types of data elements:
   a. Those that take a value within a range: for example, a payroll check amount between $ 1and $10,000 is called continuous value.
   b. Those that have a specific value: for example. Departments in a firm may be coded 100 (accounting), 110 (personnel), etc. In a data dictionary, it is described as follows:

      100 means "Accounting Department"

      101 means " Accounts Receivable Section"

      102 means " Accounts Payable Section"

      108 means " General Ledger Section"
3. Control information such as the source, date of origin, users, or access authorizations.
4. Physical location in terms of a record, file or data base.

Describing Data Structures

We describe any data structure by specifying the name of each data structure and the elements it represents, provided they are defined else- where in the data dictionary. Some elements are mandatory, whereas others are optional.

To illustrate, let us take

"BOOK- DETAILS". The data elements of this data structure are as follows:

Describing Data Flows and Data Stores

The contents of a data flow may be described by the name (s) of the data structures(s) that passes along it. In our earlier example, BOOK-DETAILS express the content of the data flow that lead to process 4. Additionally, we may specify the source of the date flow, the destination, and the volume (if any). Using the BOOK- ORDER example, data flows may be described as follows:

Data Flow	*Comments*
BOOK-DETAILS	From Newcomb Hall Bookstore (source)
AUTHOR -NAME	
TITLE OF BOOK	
EDITION	Recent edition required
QUANTITY	Minimum 40 copies

A data store is described by the data structures found in it and the data flows that feed it or are extracted from it. For example, the date store BOOK STORE- ORDER is described by the following contents:

	Comments
ORDER	Data flow/data structure feeding date store
ORDER-NUMBER	
CUSTOMER -DETAILS	Content of data store
BOOK- DETAIL	Data flow/data structure extracted from data store

Describing Processes.

This step is the logical description. We want to specify the inputs and outputs for the process and summarize the logic of the system. In constructing a data dictionary, the analyst should consider the following points:

1. Each unique data flow in the DFD must have one data dictionary entry. There is also a data dictionary entry for each data store and process.
2. Definitions must be readily accessible by name.
3. There should be no redundancy or unnecessary definitions in the data definition. It must also be simple to make updates.
4. The procedure for writing definitions should be straightforward but specific. There should be only one way of defining words.

In summary a data dictionary is an integral component of the structured specification. Without a data dictionary, the DFD lacks rigor, and without the DFD, the data dictionary is of no use. Therefore, the correlation between the two is important.

3.3 ENTITY RELATIONSHIP DIAGRAMS

Entity Relationship Diagram is a data modeling method used in software engineering to produce a conceptual data model of an information system. Diagrams created using this ER-modeling method are called Entity-Relationship Diagrams or ER diagrams or ERDs.

Notes

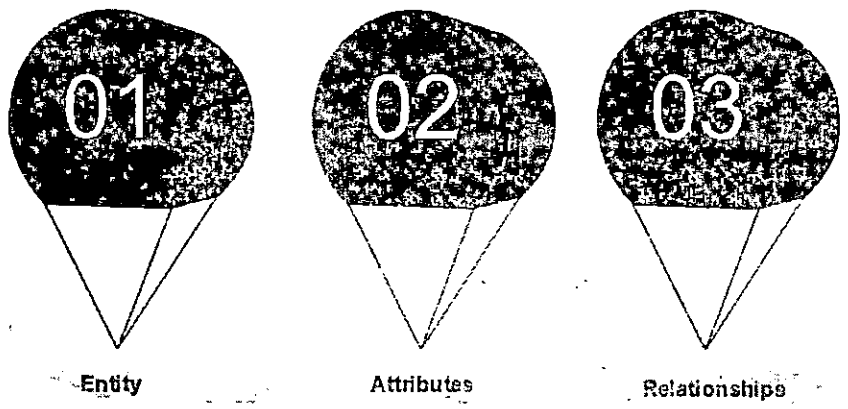
Purpose of ERD

The database analyst gains a better understanding of the data to be contained in the database through the step of constructing the ERD.

The ERD serves as a documentation tool.

Finally, the ERD is used to conriect the logical structure of the database to users. In particular, the ERD effectively communicates the logic of the database to users.

Components of an ER Diagrams



Fig,. 1 Components of an ER Diagrams

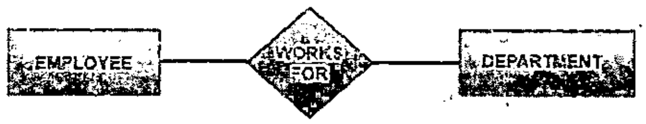
1. Entity

An entity can be a real-world object, either animate or inanimate, that can be merely identifiable. An entity is denoted as a rectangle in an ER diagram.

For example, in a school database, students, teachers, classes, and courses offered can be treated as entities. All these entities have some attributes or properties that give them their identity.

Entity Set

An entity set is a collection of related types of entities. An entity set may include entities with attribute sharing similar values. For example, a Student set may contain all the students of a school; likewise, a Teacher set may include all the teachers of a school from all faculties. Entity set need not be disjoint.

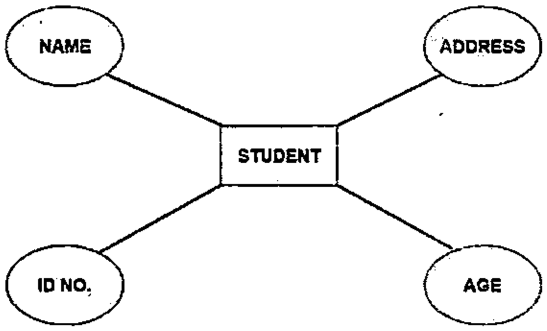


2. Attributes

Notes

Entities are denoted utilizing their properties, known as attributes. All attributes have values. For example, a student entity may have name, class, and age as attributes.

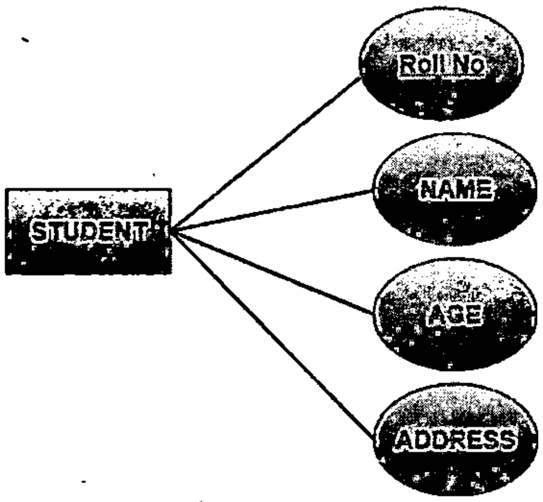
There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.



There are four types of Attributes:

- Key attribute
- Composite attribute
- Single-valued attribute
- Multi-valued attribute
- Derived attribute

1. Key attribute: Key is an attribute or collection of attributes that uniquely identifies an entity among the entity set. For example, the roll_number of a student makes him identifiable among students.



Notes

There are mainly three types of keys:

- Super key: A set of attributes that collectively identifies an entity in the entity set.
- Candidate key: A minimal super key is known as a candidate key. An entity set may have more than one candidate key.
- Primary key: A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

2. Composite attribute: An attribute that is a combination of other attributes is called a composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other characteristics such as pin code, state, country.

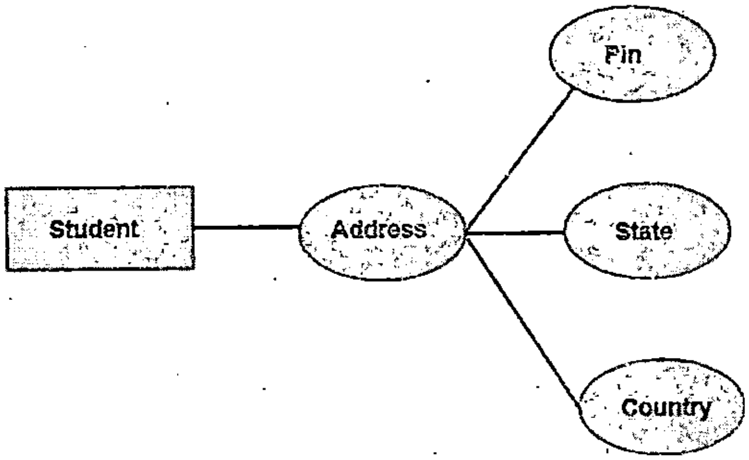
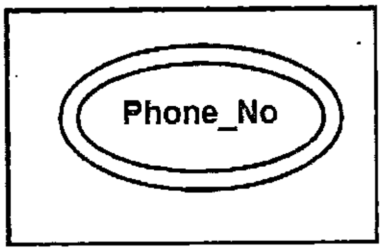
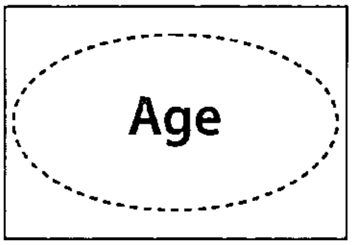


Fig. 2. Address is a composite contribute

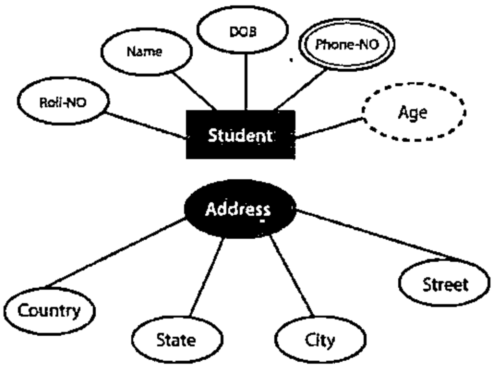
3. Single-valued attribute: Single-valued attribute contain a single value. For example, Social_Security_Number.
4. Multi-valued Attribute: If an attribute can have more than one value, it is known as a multi-valued attribute. Multi-valued attributes are depicted by the double ellipse. For example, a person can have more than one phone number, email-address, etc.



5. Derived attribute: Derived attributes are the attribute that does not exist in the physical database, but their values are derived from other attributes present in the database. For example, age can be derived from date_of_birth. In the ER diagram, Derived attributes are depicted by the dashed ellipse.



The complete entity type student with its attributes can be represented as:



3. Relationships

The association among entities is known as relationship. Relationships are represented by the diamond-shaped box. For example, an employee works_at a department, a student enrolls in a course. Here, Works_at and Enrolls are called relationships.

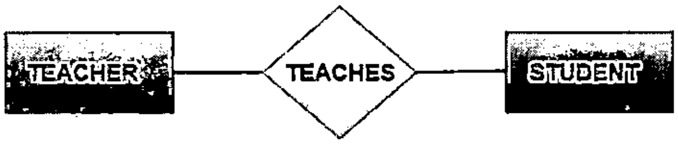


Fig. 3. Relationship in ERD

Relationship set

A set of relationships of a similar type is known as a relationship set. Like entities, a relationship too can have attributes. These attributes are called descriptive attributes.

Notes

Degree of a relationship set

The number of participating entities in a relationship describes the degree of the relationship. The three most common relationships in E-R models are:

- Unary (degree1)
- Binary (degree2)
- Ternary (degree3)

1. Unary relationship: This is also called recursive relationships. It is a relationship between the instances of one entity type. For example, one person is married to only one person.

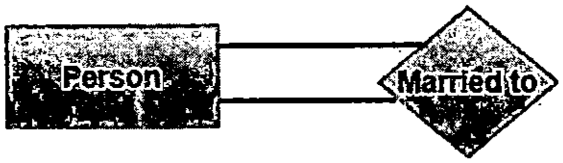


Fig. 4. Unary relationship

2. Binary relationship: It is a relationship between the instances of two entity types. For example, the Teacher teaches the subject.

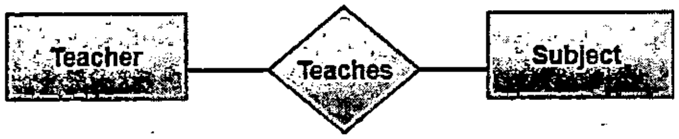


Fig. 5. Binary relationship

3. Ternary relationship: It is a relationship amongst instances of three entity types. In fig, the relationships "may have" provide the association of three entities, i.e., TEACHER, STUDENT, and SUBJECT. All three entities are many-to-many participants. There may be one or many participants in a ternary relationship.

In general, "n" entities can be related by the same relationship and is known as n-ary relationship.

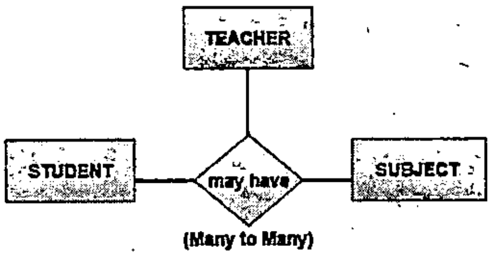


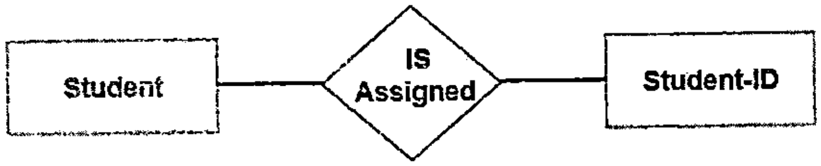
Fig. 6. Ternary relationship

Cardinality

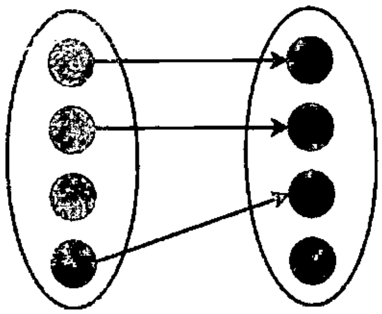
Cardinality describes the number of entities in one entity set, which can be associated with the number of entities of other sets via relationship set.

Types of Cardinalities

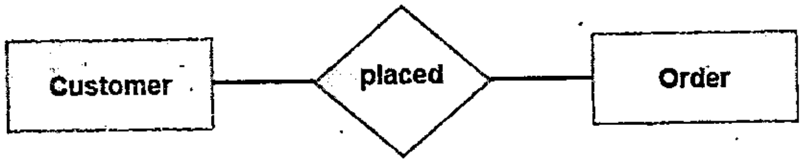
1. **One to One:** One entity from entity set A can be contained with at most one entity of entity set B and vice versa. Let us assume that each student has only one student ID, and each student ID is assigned to only one person. So, the relationship will be one to one.



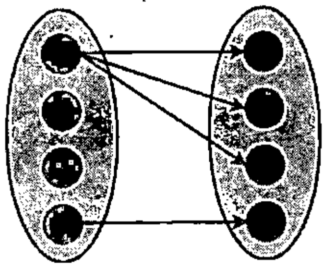
Using Sets, it can be represented as:



2. **One to many:** When a single instance of an entity is associated with more than one instances of another entity then it is called one to many relationships. For example, a client can place many orders; a order cannot be placed by many customers.

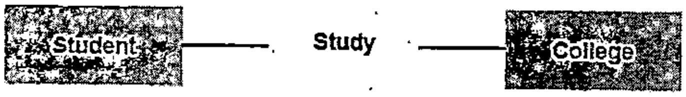


Using Sets, it can be represented as:

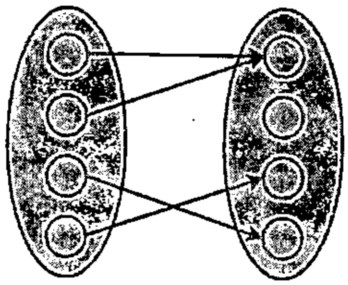


Notes

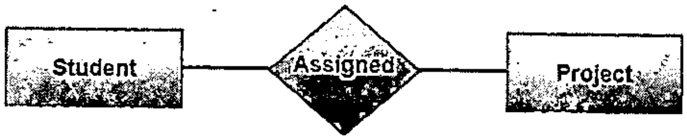
3. **Many to One:** More than one entity from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A. For example - many students can study in a single college, but a student cannot study in many colleges at the same time.



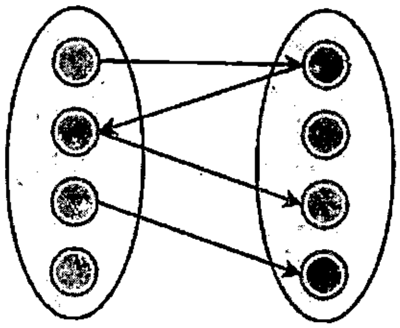
Using Sets, it can be represented as:



4. **Many to Many:** One entity from A can be associated with more than one entity from B and vice-versa. For example, the student can be assigned to many projects, and a project can be assigned to many students.



Using Sets, it can be represented as:



3.4 DATA DICTIONARIES

Data Dictionary is the major component in the structured analysis model of the system. It lists all the data items appearing in DFD. A data dictionary in Software Engineering means a file or a set of files that includes a database's metadata (hold records about other objects in the database), like data ownership, relationships of the data to another object, and some other data.

Example a data dictionary entry: GrossPay = regular pay + overtime pay

Case Tools is used to maintain data dictionary as it captures the data items appearing in a DFD automatically to generate the data dictionary.

Components of Data Dictionary:

In Software Engineering, the data dictionary contains the following information:

- Name of the item: It can be your choice.
- Aliases: It represents another name.
- Description: Description of what the actual text is all about.
- Related data items: with other data items.
- Range of values: It will represent all possible answers.

Data Dictionary Notations tables :

The Notations used within the data dictionary are given in the table below as follows:

Notation

Notations	*Meaning*
X = a+b	X consists data elements a and b.
X = [a/b]	X consists of either elements a or b.
X = a X	X consists of optimal data elements a.
X = y[a]	X consists of y or more events of data element a
X = [a] z	X consists of z or less events of data element a
X = y [a] z	X consists of some events of data elements between y and z.

Features of Data Dictionary:

Here, we will discuss some features of the data dictionary as follows.

- It helps in designing test cases and designing the software.
- It is very important for creating an order list from a subset of the items list.
- It is very important for creating an order list from a complete items list.
- The data dictionary is also important to find the specific data item object from the list.

Uses of Data Dictionary :

Here, we will discuss some use cases of the data dictionary as follows.

- Used for creating the ordered list of data items
- Used for creating the ordered list of a subset of the data items
- Used for Designing and testing software in Software Engineering
- Used for finding data items from a description in Software Engineering

Importance of Data Dictionary:

- It provides developers with standard terminology for all data.
- It provides developers to use different terms to refer to the same data.
- It provides definitions for different data
- Query handling is facilitated if a data dictionary is used in RDMS.

3.5 PROCESS MODELING

Process modeling is an analytical procedure to breakdown work processes to improve their efficiency.

Process analysis is the rational segmentation of business processes into different phases to understand them and look for ways to improve their overall efficiency. It refers to their full-fledged analysis, which also includes a series of logically articulated routine tasks that utilize organizational resources to transform a work process with the aim of benefiting the process and maintaining its excellence.

In various software engineering methods, the approach taken for requirement engineering often includes comprehensive modeling of distinct aspects such as data, system structure, or behavior. These models are the basis from which system design and implementation are contrived in the later stages of the development process and are the flagship mode of communication between expert users and system developers.

Because requirement specifications' quality is a determining factor for correction costs and software quality, a huge chunk of the effort is dedicated to system modeling in software development's early stages. It is for these reasons that organizations often set down process models describing the workflows or processes of a business in a graphic format to accomplish a specific goal.

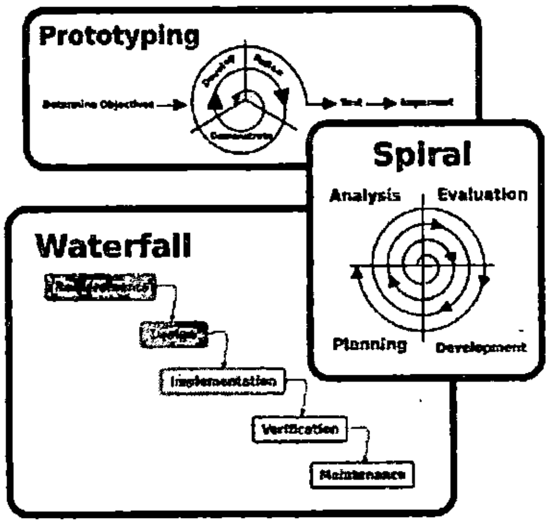
Software processes are the activities for designing, implementing, and testing a software system. The software development process is complicated and involves a lot more than technical knowledge.

That's where software process models come in handy. A software process model is an abstract representation of the development process.

In this section, we will introduce you to the top seven software process models and discuss when to use them. A software process model is an abstraction of the software development process. The models specify the stages and order of a process. So, think of this as a representation of the order of activities of the process and the sequence in which they are performed.

A model will define the following:

- The tasks to be performed
- The input and output of each task
- The pre and post conditions for each task
- The flow and sequence of each task
- The goal of a software process model is to provide guidance for controlling and coordinating the tasks to achieve the end product and objectives as effectively as possible.



There are many kinds of process models for meeting different requirements. We refer to these as SDLC models (Software Development Life Cycle models). The most popular and important SDLC models are as follows:

- Waterfall model
- V model
- Incremental model
- RAD model
- Agile model
- Iterative model
- Prototype model
- Spiral model

Factors in Choosing a Software Process

Choosing the right software process model for your project can be difficult. If you know your requirements well, it will be easier to select a model that best matches your needs. You need to keep the following factors in mind when selecting your software process model:

Project requirements

Before you choose a model, take some time to go through the project requirements and clarify them alongside your organization's or team's expectations. Will the user need to specify requirements in detail after each iterative session? Will the requirements change during the development process?

Project size

Consider the size of the project you will be working on. Larger projects mean bigger teams, so you'll need more extensive and elaborate project management plans.

Project complexity

Complex projects may not have clear requirements. The requirements may change often, and the cost of delay is high. Ask yourself if the project requires constant monitoring or feedback from the client.

Cost of delay

Is the project highly time-bound with a huge cost of delay, or are the timelines flexible?

Customer involvement

Do you need to consult the customers during the process? Does the user need to participate in all phases?

Familiarity with technology

This involves the developers' knowledge and experience with the project domain, software tools, language, and methods needed for development.

Project resources

This involves the amount and availability of funds, staff, and other resources.

Types of Software Process Models

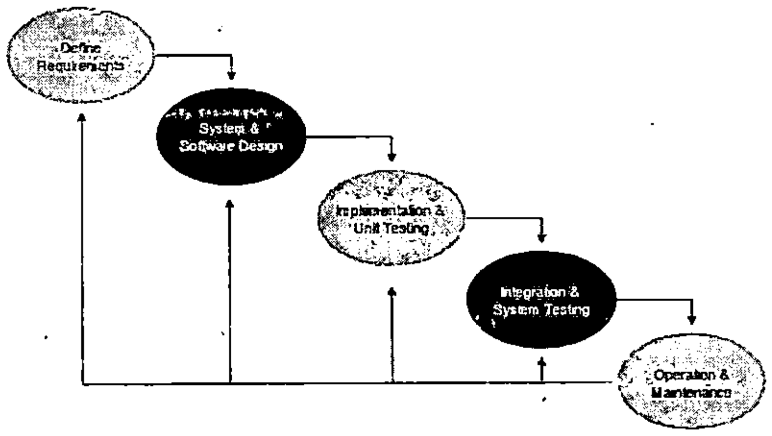
As we mentioned before, there are multiple kinds of software process models that each meet different requirements. Below, we will look at the top seven types of software process models that you should know.

Waterfall Model

The waterfall model is a sequential, plan driven-process where you must plan and schedule all your activities before starting the project. Each activity in the waterfall model is represented as a separate phase arranged in linear order.

It has the following phases:

- Requirements
- Design
- Implementation
- Testing
- Deployment
- Maintenance



Each of these phases produces one or more documents that need to be approved before the next phase begins. However, in practice, these phases are very likely to overlap and may feed information to one another.

The software process isn't linear, so the documents produced may need to be modified to reflect changes.

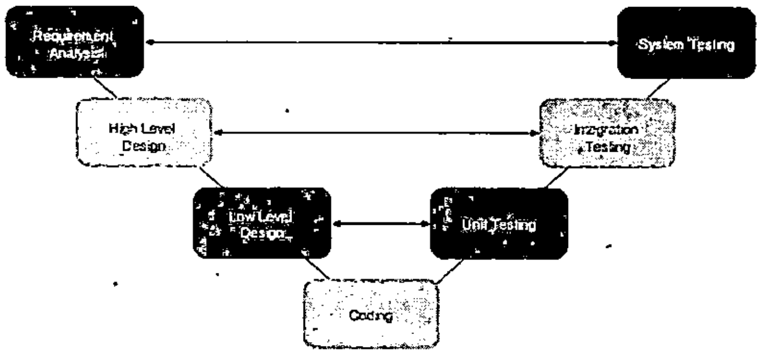
The waterfall model is easy to understand and follow. It doesn't require a lot of customer involvement after the specification is done. Since it's inflexible, it can't adapt to changes. There is no way to see or try the software until the last phase.

The waterfall model has a rigid structure, so it should be used in cases where the requirements are understood completely and unlikely to radically change..

V Model

The V model (Verification and Validation model) is an extension of the waterfall model. All the requirements are gathered at the start and cannot be changed. You have a corresponding testing activity for each stage. For every phase in the development cycle, there is an associated testing phase.

Notes



The V model is highly disciplined, easy to understand, and makes project management easier. But it isn't good for complex projects or projects that have unclear or changing requirements. This makes the V model a good choice for software where downtimes and failures are unacceptable.

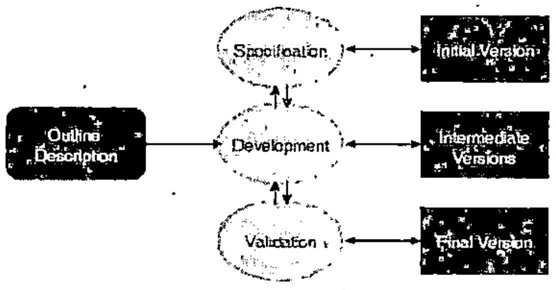
Incremental Model

The incremental model divides the system's functionality into small increments that are delivered one after the other in quick succession. The most important functionality is implemented in the initial increments.

The subsequent increments expand on the previous ones until everything has been updated and implemented.

Incremental development is based on developing an initial implementation, exposing it to user feedback, and evolving it through new versions. The process' activities are interwoven by feedback.

Each iteration passes through the requirements, design, coding, and testing stages.



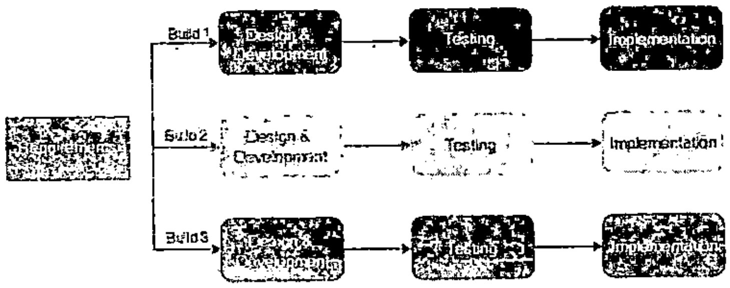
The incremental model lets stakeholders and developers see results with the first increment. If the stakeholders don't like anything, everyone finds out a lot sooner. It is efficient as the developers only focus on what is important and bugs are fixed as they arise, but you need a clear and complete definition of the whole system before you start.

The incremental model is great for projects that have loosely-coupled parts and projects with complete and clear requirements.

Iterative Model

The iterative development model develops a system through building small portions of all the features. This helps to meet initial scope quickly and release it for feedback.

In the iterative model, you start off by implementing a small set of the software requirements. These are then enhanced iteratively in the evolving versions until the system is completed. This process model starts with part of the software, which is then implemented and reviewed to identify further requirements.



Like the incremental model, the iterative model allows you to see the results at the early stages of development. This makes it easy to identify and fix any functional or design flaws. It also makes it easier to manage risk and change requirements.

The deadline and budget may change throughout the development process, especially for large complex projects. The iterative model is a good choice for large software that can be easily broken down into modules.

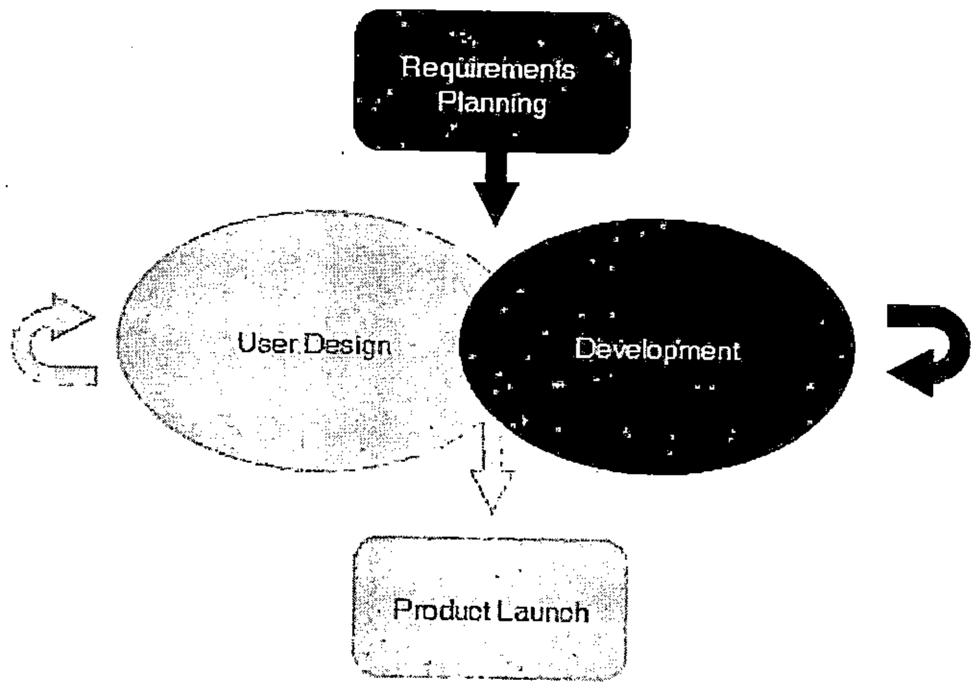
RAD Model

The Rapid Application Development (RAD model) is based on iterative development and prototyping with little planning involved. You develop functional modules in parallel for faster product delivery. It involves the following phases:

- Business modeling
- Data modeling
- Process modeling
- Application generation
- Testing and turnover

The RAD concept focuses on gathering requirements using focus groups and workshops, reusing software components, and informal communication.

Notes



The RAD model accommodates changing requirements, reduces development time, and increases the reusability of components. But it can be complex to manage. Therefore, the RAD model is great for systems that need to be produced in a short time and have known requirements.

Spiral Model

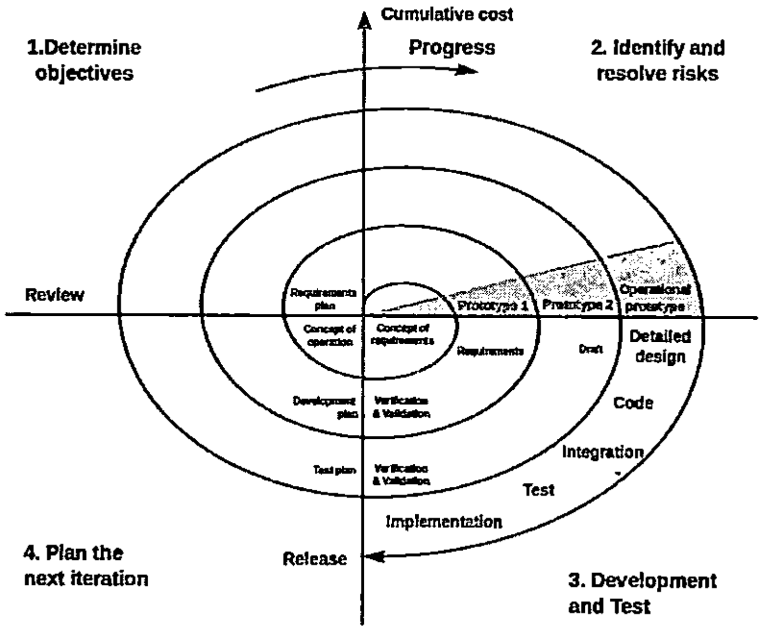
The spiral model is a risk driven iterative software process model. The spiral model delivers projects in loops. Unlike other process models, its steps aren't activities but phases for addressing whatever problem has the greatest risk of causing a failure.

It was designed to include the best features from the waterfall and introduces risk-assessment.

You have the following phases for each cycle:

- Address the highest-risk problem and determine the objective and alternate solutions
- Evaluate the alternatives and identify the risks involved and possible solutions
- Develop a solution and verify if it's acceptable
- Plan for the next cycle

You develop the concept in the first few cycles, and then it evolves into an implementation. Though this model is great for managing uncertainty, it can be difficult to have stable documentation. The spiral model can be used for projects with unclear needs or projects still in research and development.



Agile model

The agile process model encourages continuous iterations of development and testing. Each incremental part is developed over an iteration, and each iteration is designed to be small and manageable so it can be completed within a few weeks.

Each iteration focuses on implementing a small set of features completely. It involves customers in the development process and minimizes documentation by using informal communication.

Agile development considers the following:

- Requirements are assumed to change
- The system evolves over a series of short iterations
- Customers are involved during each iteration
- Documentation is done only when needed

Though agile provides a very realistic approach to software development, it isn't great for complex projects. It can also present challenges during transfers as there is very little documentation. Agile is great for projects with changing requirements.

Some commonly used agile methodologies include:

- Scrum: One of the most popular agile models, Scrum consists of iterations called sprints. Each sprint is between 2 to 4 weeks long and is preceded by planning. You cannot make changes after the sprint activities have been defined.

Notes

- Extreme Programming (XP): With Extreme Programming, an iteration can last between 1 to 2 weeks. XP uses pair programming, continuous integration, test-driven development and test automation, small releases, and simple software design.
- Kanban: Kanban focuses on visualizations, and if any iterations are used they are kept very short. You use the Kanban Board that has a clear representation of all project activities and their numbers, responsible people, and progress.

3.6 STRUCTURED ENGLISH

Structured English is the use of the English language with the syntax of structured programming to communicate the design of a computer program to non-technical users by breaking it down into logical steps using straightforward English words. Structured English gives aims to get the benefits of both the programming logic and natural language: program logic helps to attain precision, whilst natural language helps with the familiarity of the spoken word.

It is the basis of some programming languages such as SQL (Structured Query Language) "for use by people who have need for interaction with a large database but who are not trained programmers"

Structured English is a narrative form of English written as a series of blocks that use indentation and capitalization to represent a hierarchical structure of logic specifications. This method does not show any decisions or rules, but it states the rules and is used when an individual or an organization is trying to overcome the problems of an ambiguous language by stating the actions and conditions used when making decisions and formulating procedures.

Structured English is based on structured logic; it is used when process logic involves formulas or iteration, or when structured decisions are not too complex. Structured English is used to express all logic in terms of sequential structures, decision structures, iterations and case structures. This modified form of English is used to specify the logic of information processes by using a subset of English vocabulary to express process procedures.

Structured English derives from structured programming and its use of logical construction and imperative statements. This process is designed to carry out instructions for actions by creating decision statements that use structured programming terms, such as "IF", "ELSE" and "THEN".

Structure statements are developed and defined by using the following types of statements:

- Sequence Structure: This is the single step or action included in the sequence process; it does not depend on the existence of other conditions. If the sequence structure does encounter a condition, it is taken into consideration.
- Decision Structure: This occurs when two or more actions rely on the value of a specific condition. The condition is expanded and the necessary decisions are made.
- Iteration/Repetition Structure: Certain conditions will only occur after specific conditions are executed. Iterative instructions help an analyst describe these specific cases.

Notes

3.7 DECISION TREES

Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

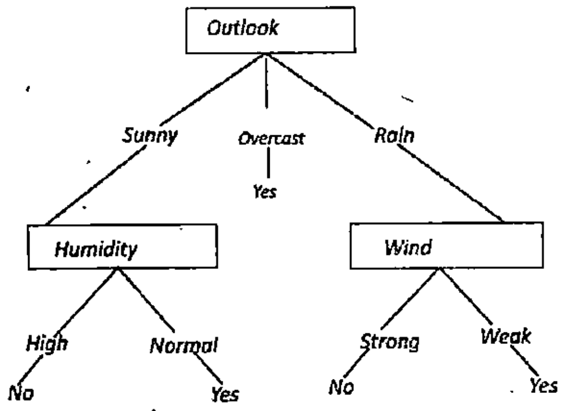


Fig. 7. A decision tree for the concept PlayTennis.

Construction of Decision Tree

A tree can be "learned" by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions.

The construction of decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. In general decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification.

Decision Tree Representation

Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute as shown in the above figure. This process is then repeated for the subtree rooted at the new node.

The decision tree in above figure classifies a particular morning according to whether it is suitable for playing tennis and returning the classification associated with the particular leaf.(in this case Yes or No).

Notes

For example, the instance

(Outlook = Rain, Temperature = Hot, Humidity = High, Wind = Strong)

would be sorted down the leftmost branch of this decision tree and would therefore be classified as a negative instance.

In other words we can say that decision tree represent a disjunction of conjunctions of constraints on the attribute values of instances.

(Outlook = Sunny ^ Humidity = Normal) v (Outlook = Overcast) v (Outlook = Rain ^ Wind = Weak)

Strengths and Weakness of Decision Tree approach

The strengths of decision tree methods are:

- Decision trees are able to generate understandable rules.
- Decision trees perform classification without requiring much computation.
- Decision trees are able to handle both continuous and categorical variables.
- Decision trees provide a clear indication of which fields are most important for prediction or classification.

The weaknesses of decision tree methods :

- Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.
- Decision trees are prone to errors in classification problems with many class and relatively small number of training examples.

Decision tree can be computationally expensive to train. The process of growing a decision tree is computationally expensive. At each node, each candidate splitting field must be sorted before its best split can be found.

In some algorithms, combinations of fields are used and a search must be made for optimal combining weights. Pruning algorithms can also be expensive since many candidate sub-trees must be formed and compared

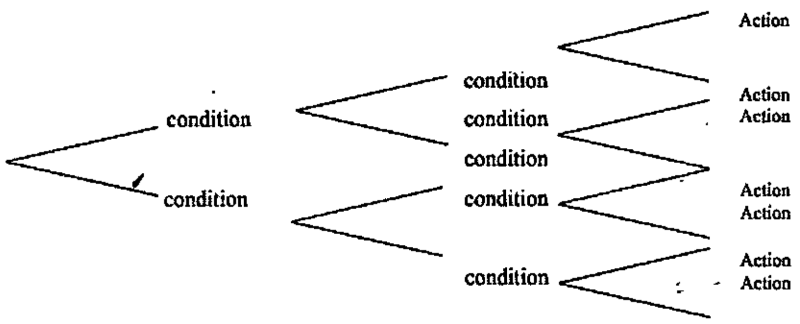
As you know well, people often have different ways of saying the same thing. For example, the discount conditions discussed in the last example can also be stated in the following ways:

1. Greater than $10,000, grater than or equal $ 5,000 but less than or equal to $ 10,000, and below $5,000
2. Not less than $10,000, not more than $ 10,000 but at least $ 5,000, and not $5,000 or more

Having different ways of saying the same thing can create difficulties in communication during systems studies (analyst and manager may misunderstand each other's comments or forget to discuss all the details).

Thus, analysts seek to prevent misunderstandings. They also need to organize information collected about decision making. Decision trees are one of the methods for describing decisions, while avoiding difficulties in communication

CONDITION	ACTION
Order is signed	Begin order verification process
Order is unsigned	Begin merchandise acceptance processing.
CONDITION	ACTION
Size of order : Over $ 10,000	Take 3 % discount form invoice total
$5,000 to $10,000	Take 2 % discount form invoice
Less than $5,000	Pay full invoice amount.



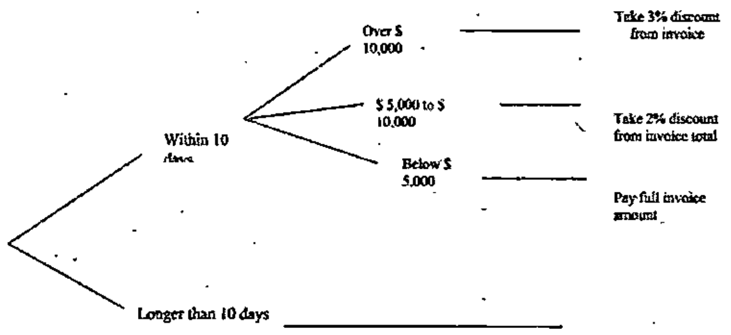
3.8 DECISION - TREE CHARACTERISTICS

A decision tree is a diagram that presents conditions and actions sequentially and thus shows which conditions to consider first, which second, and so on. It is also a method of showing the relationship of each condition and its permissible actions. The diagram resembles branches on a tree, hence the name.

The root of the tree, on the left of the diagram, is the starting point of the decision sequence. The particular branch to be followed depends on the conditions that exist and the decision to be made. Progression from left to right along a particular branch is the result of making a series of decisions.

Following each decision points is the next set of decision to be considered. The nodes of the tree thus represent conditions and indicate that a determination must be made about which condition exists before the next path can be chosen. The right side of the tree lists the actions to be taken depending on the sequence of conditions that is followed.

Notes



Using Decision Trees

Developing decision trees is beneficial to analysts in two ways. First of all, the need to describe conditions and actions forces analysts to formally identify the actual decision that must be made. It becomes difficult for them to overlook and integral step in the decision process, whether it depends on quantitative or nonquantitative variables.

It is possible, for instance, to show what discount action to take, depending on the number of dollar spent by customers. When an organization opens accounts with dealers and suppliers, it formalizes an agreements for taking discounts from the full invoice price. Two conditions are specified in this agreement: first, the invoice must always be paid within ten days of its receipt, and, second, the size of the discount will depend on the value of the invoice. It is agreed that under some conditions the organization can take the action of deducting a 3 percent discount; under other conditions, a 2 percent discount; and under all other conditions, no discount is allowed.

Decision trees also force analysts to consider the sequence of decisions. Consider the sequence of decision in the example, you can quickly determine that one condition is amount of the invoice-does not matter unless another condition is met and the invoice is paid within the time established by the supplier – ten days.

The other conditions are relevant only if that condition is true. Therefore, the decision tree identifies the time condition first and shows two values (within ten days and longer than ten days). The discount condition is described next, but only for the branch of the tree for WITHIN TEN DAYS.

The LONGER THAN TEN DAYS branch has no other relevant conditions and so show the resulting action (unconditionally). This tree shows that the action PAY FULL INVOICE AMOUNT applies under two different conditions. It also shows implicitly that there is no reason to pay invoice of less than $5,000 within ten days, since there is no discount available for these amounts.

The decision tree shows the nonquantitative conditions for processing accounts payable: signed invoice, authorized purchase, and correct pricing. Depending on the set of conditions that exist, one of two actions can be taken: payment can be authorized or the submitted invoice can be rejected.

Notice how clearly each alternative is shown in the decision tree. Sometimes in more complex business situations, the specific action most appropriate under several conditions is not readily apparent without formal analysis of this nature.

Analysts find that in accounts payable processing, it is necessary to determine whether a purchase order is available and valid and whether the invoice is processed properly before it is actually paid.

In turn, they must learn of the conditions for proper invoice processing. Full development and examination of the decision tree also shows clearly that there are only two ways an invoice can be authorized for payment but many conditions under which payment can be rejected.

The sequence of decision is easy to see in this example. The condition of having a valid purchase order does not matter unless the invoice has been signed. The signature is important, since the condition of having a signed invoice must be met before processing can continue. Then analysts can consider the authorization condition.

Identifying Data Requirements

We have already pointed out the use of decision trees to formally highlight the sequential nature of many business decision, and we have shown that decision trees are effective when describing business problems of more than one dimension or condition.

However, they also identify critical data requirements surrounding the decision process; that is, they indicate the sets of data the manager requires to formulate decision or select action. The explicit data in the payment example are payment data, amount of invoice, and discount allowance percentage.

There are other important data elements such as invoice details (number, supplier name and address), new invoice amount payable, and adjustments to "discount taken" that are indict (not directly expressed in the decision tree). The analyst must identify and list all the data used in the decision process, even must identify and list all the data used in the decision process, even though the decision tree does not show all the individual data items.

It decision trees are assembled after the completion of data flow analysis (which tracks the flow of data through busbies processes), critical data may already be defined in the data dictionary (which describes system data and where they are used).

If decision trees are identify each data element needed to make a decision trees are identify each data element needed to make a decision. The data dictionary format, is useful for listing and describing data elements as they are identified and understood.

The date requirements discussed for decision trees also apply to the other decision – analysis methods that will be discussed. Analysis's need to describe and define all data used in decision making, so that the system can be designed to produce data properly.

Avoiding problems with Decision Trees

Decision trees may not always be the best tolls for decision analysis. A decision tree for a complex system with many sequences of steps and combinations of conditions will be unwieldy. A large number of branches with many paths through them will could rather than aid analysis. The analyst may not be able to determine which business policies or

Notes

practices guide the formulation of specific decisions. Where these problems arise, decision tables should be considered.

3.9 DECISION TABLES

A decision table is a brief visual representation for specifying which actions to perform depending on given conditions. The information represented in decision tables can also be represented as decision trees or in a programming language using if-then-else and switch-case statements.

A decision table is a good way to settle with different combination inputs with their corresponding outputs and is also called a cause-effect table. The reason to call cause-effect table is a related logical diagramming technique called cause-effect graphing that is basically used to obtain the decision table.

Importance of Decision Table

Decision tables are very much helpful in test design techniques.

- It helps testers to search the effects of combinations of different inputs and other software states that must correctly implement business rules.
- It provides a regular way of starting complex business rules, that is helpful for developers as well as for testers.
- It assists in the development process with the developer to do a better job. Testing with all combinations might be impractical.
- A decision table is basically an outstanding technique used in both testing and requirements management.
- It is a structured exercise to prepare requirements when dealing with complex business rules.
- It is also used in model complicated logic.

Decision Table in test designing:

Blank Decision Table

CONDITIONS	STEP 1	STEP 2	STEP 3	STEP 4
Condition 1				
Condition 2				
Condition 3				
Condition 4				

Notes

Decision Table: Combinations

CONDITIONS	STEP 1	STEP 2	STEP 3	STEP 4
Condition 1	Y	Y	N	N
Condition 2	Y	N	Y	N
Condition 3	Y	N	N	Y
Condition 1	N	Y	Y	N

Advantage of Decision Table:

- Any complex business flow can be easily converted into test scenarios & test cases using this technique.
- Decision tables work iteratively which means the table created at the first iteration is used as input tables for the next tables. The iteration is done only if the initial table is not satisfactory.
- Simple to understand and everyone can use this method to design the test scenarios & test cases.
- It provides complete coverage of test cases which helps to reduce the rework on writing test scenarios & test cases.
- These tables guarantee that we consider every possible combination of condition values. This is known as its completeness property.

FIGURE Structured English- Using Data Dictionary Values.

```
FIGURE Structured English- Using Data Dictionary Values.
COMPUTE-DISCOUNT
Add up the number of copies per book title
IF order is from bookstore
And – IF ORDER -SIZE is SMALL
THEN: Discount is 25%
ELSE (ORDER -SIZE is MINIMUM)
 So: no discount is allowed
ELSE (order is from libraries or individual customers)
So-IF ORDER -SIZE is LARGE
 Discount is 15%
ELSE IF OREDR SIZE is EMDIUM
 Discount is 10%
ELSE IF ORDER-SIZE is SMALL
 Discount is 5%
ELSE (ORDER -SIZE is MINIMUM)
 So: no discount is allowed
```

A major drawback of a decision tree is the lack of information in its format to tell us what other combinations of conditions to test. This is where the decision table is useful. A

Notes

decision table is a table of contingencies for defining a problem and the actions to be taken. It is single representation of the relationships between conditions and actions.

The stub part is divided into an upper quadrant called the condition stub and a lower quadrant called the action stub. The entry part is also divided into an upper quadrant, called the condition entry and a lower quadrant called the action entry. The four elements and their definitions are summarized in Figure.

The stub part is divided into an upper quadrant called the condition stub and a lower quadrant called the action stub. The entry part is also divided into an upper quadrant, called the condition entry and a lower quadrant called the action entry. The four elements and their definitions are summarized in Figure.

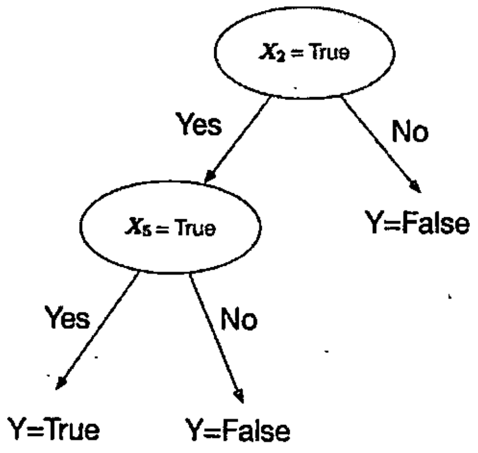


Fig. 8. Elements and Definitions in a Decision Table

The answers are represented by a Y to signify yes, an N to signify no, or a blank to show that the condition involved has not been tested. In the action entry quadrant an X(or a check mark will do) indicates the response to the answer(s) entered in the condition entry quadrant. Furthermore, each column represents a decision or a rule. For example, rule 1 states:

IF customer is a bookstore and order size is 6 copies or more. THEN allow 25% discount So, according to the decision table, we have six decisions and therefore six rules. A look at the table provides each decision (answer) immediately the following rules should be followed in constructing decision tables:

1. A decision should be given a name, shown in the top left of the table.
2. The logic of the decision table is independent of the sequence in which the condition rules are written, but the action takes place in the order in which the events occur.
3. Standardized language must be used consistently.
4. Duplication of terms or meanings should be eliminated, where possible.

Pros and Cons of Each Tool

Which tool is the best depends on a number of factors: the nature and complexity of the problem the number of actions resulting from the decision, and the ease of use. In reviewing the benefits and limitations of each tool, we come to the following conclusion:

1. The primary strength of the DFD is its ability to represent data flows. It may beused at high or low level of analysis and provides good system documentation. However, the tool only weakly shows input and output detail. The user often finds it confusing initially.
2. The data dictionary helps the analyst simplify the structure for meeting the data requirements of the system. It may be used at high or low levels of analysis, but it does not provide functional details, and it is not acceptable to many nontechnical users.
3. Structured English is best used when the problem requires sequences of actions with decisions.
4. Decision trees are sued to verify logic and in problems that involve a few complex decisions resulting in limited number of actions.
5. Decision trees and decision tables are best suited for dealing with complex branching routines such as calculating discounts or sales commissions or inventory control procedures.

Given the pros and cons of structured tools, the analyst should be trained in the sue of various tools for analysis and design He/She should use decision table and structured English to get to the heart of complex problems. A decision table is perhaps the most useful tool for communicating problem details to the user.

The major contribution of structured analysis to the system development life cycle is producing a definable and measurable document – the structured specification. Other benefits include increased user involvement, improved communication between user and designer, reduction of total personnel time, and fewer " kinks" during detailed design and implementation. The only drawback is increased analyst and user time in the process. Overall the benefits outweigh the drawbacks, which make-structured analysis tools viable alternatives in system development.

3.10 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis (OOA) is the first technical activity performed as part of object oriented software engineering. OOA is process of discovery where a development team understands and models the requirements of the system. In OOA requirements are organized as objects. It integrates all the process and data. But in others or traditional structural analysis both process and data are considered independently/separately. They use flow chart/structure charts for process and ER diagrams for data.

But In OOA some advance models are used. The common models used in OOA are: Use cases, Object models. Use cases describe pictures or overview for standard domain functions that the system must achieved. Object models describe the names, class relations, operations, and properties of the main objects. User-interface prototypes can also be created for better understanding.

Object Oriented Analysis (OOA) begins by looking at the problem domain (the area of expertise or application that needs to analyze in order to solve a problem). Its aim is to produce a conceptual model of the information that exists in the area being analyzed. For the analysis there are a variety of sources. It can be a formal document, a written requirements statement, interviews with stakeholders/other interested parties, other methods, etc. The final result of object oriented analysis will appear in the form of a conceptual model that describes what the system is functionally required to do.

Functionalities of OOA

The core activities in OOA are given below:

- Find the objects
- Organize the objects by creating object model diagram
- Explain how the objects communicates with each others
- Set the characteristic or behavior of the objects
- Set the internal of the objects

Advantages of OOA

The OOA provides better performance. Some common advantages of OOA are given bellow:

- Its focuses on data rather than the procedures as in Structured Analysis
- The objectives of encapsulation and data hiding help the developer to develop the systems that cannot be tampered by other parts of the system
- It allows effective software complexity management by the virtue of modularity
- It can be upgraded from small to large system easily

Structured Analysis VS Object Oriented Analysis

Some differences between Structured Analysis and Object Oriented Analysis are given bellow:

- Structured Analysis treats processes and data as separate components. Where, OO Analysis Combines processes and data into single component which called object
- Structured Analysis does not support re-usability of code. So, the development time and cost is naturally high. But OO analysis supports code re-usability which reduce the development time and cost
- The Object Oriented Analysis and Design (OOAD) is a new dimension to develop the software system. It provides better performance comparatively to tradition structured analysis. But in some case it is inappropriate. We should consider the advantages and disadvantages of both OOA and structured analysis to choice correct techniques.

OOA introduces new concepts to investigate a problem. It is based in a set of basic principles, which are as follows:

- The information domain is modeled.
- Behavior is represented.
- Function is described.

- Data, functional, and behavioral models are divided to uncover greater detail.
- Early models represent the essence of the problem, while later ones provide implementation
- details.

The above notes principles form the foundation for the OOA approach.

Let us discuss in detail.

In software engineering, a domain model is a conceptual model of the domain that incorporates both behavior and data. In ontology engineering, a domain model is a formal representation of a knowledge domain with concepts, roles, datatypes, individuals, and rules, typically grounded in a description logic.

A domain model is a system of abstractions that describes selected aspects of a sphere of knowledge, influence or activity (a domain[3]). The model can then be used to solve problems related to that domain. The domain model is a representation of meaningful real-world concepts pertinent to the domain that need to be modeled in software. The concepts include the data involved in the business and rules the business uses in relation to that data. A domain model leverages natural language of the domain.

A domain model generally uses the vocabulary of the domain, thus allowing a representation of the model to be communicated to non-technical stakeholders. It should not refer to any technical implementations such as databases or software components that are being designed.

Usage

A domain model is generally implemented as an object model within a layer that uses a lower-level layer for persistence and "publishes" an API to a higher-level layer to gain access to the data and behavior of the model.

In the Unified Modeling Language (UML), a class diagram is used to represent the domain model.

3.11 THE CONCEPT OF OBJECT-ORIENTATION

- Object-orientation is what's referred to as a programming paradigm. It's not a language itself but a set of concepts that is supported by many languages.
- If you aren't familiar with the concepts of object-orientation, you may take a look at The Story of Object-Oriented Programming.
- If everything we do in these languages is object-oriented, it means, we are oriented or focused around objects.
- Now in an object-oriented language, this one large program will instead be split apart into self contained objects, almost like having several mini-programs, each object representing a different part of the application.
- And each object contains its own data and its own logic, and they communicate between themselves.

Notes

- These objects aren't random. They represent the way you talk and think about the problem you are trying to solve in your real life.
- They represent things like employees, images, bank accounts, spaceships, asteroids, video segments, audio files, or whatever exists in your program.

3.12 OBJECT-ORIENTED ANALYSIS AND DESIGN (OOAD)

It's a structured method for analyzing, designing a system by applying the object-orientated concepts, and developing a set of graphical system models during the development life cycle of the software.

OOAD In The SDLC

The software life cycle is typically divided up into stages going from abstract descriptions of the problem to designs then to code and testing and finally to deployment.

Object-Oriented Analysis And Design (OOAD) process

- The earliest stages of this process are analysis (requirements) and design.
- The distinction between analysis and design is often described as "what Vs how".
- In analysis developers work with users and domain experts to define what the system is supposed to do. Implementation details are supposed to be mostly or totally ignored at this phase.
- The goal of the analysis phase is to create a model of the system regardless of constraints such as appropriate technology. This is typically done via use cases and abstract definition of the most important objects using conceptual model.
- The design phase refines the analysis model and applies the needed technology and other implementation constrains.
- It focuses on describing the objects, their attributes, behavior, and interactions. The design model should have all the details required so that programmers can implement the design in code.
- They're best conducted in an iterative and incremental software methodologies. So, the activities of OOAD and the developed models aren't done once, we will revisit and refine these steps continually.

Object-Oriented Analysis

In the object-oriented analysis, we ...

- Elicit requirements: Define what does the software need to do, and what's the problem the software trying to solve.
- Specify requirements: Describe the requirements, usually, using use cases (and scenarios) or user stories.

- Conceptual model: Identify the important objects, refine them, and define their relationships and behavior and draw them in a simple diagram.
- We're not going to cover the first two activities, just the last one. These are already explained in detail in Requirements Engineering.

Object-Oriented Design

- The analysis phase identifies the objects, their relationship, and behavior using the conceptual model (an abstract definition for the objects).
- While in design phase, we describe these objects (by creating class diagram from conceptual diagram — usually mapping conceptual model to class diagram), their attributes, behavior, and interactions.
- In addition to applying the software design principles and patterns which will be covered in later tutorials.
- The input for object-oriented design is provided by the output of object-oriented analysis. But, analysis and design may occur in parallel, and the results of one activity can be used by the other.

In the object-oriented design, we ...

- Describe the classes and their relationships using class diagram.
- Describe the interaction between the objects using sequence diagram.
- Apply software design principles and design patterns.
- A class diagram gives a visual representation of the classes you need. And here is where you get to be really specific about object-oriented principles like inheritance and polymorphism.
- Describing the interactions between those objects lets you better understand the responsibilities of the different objects, the behaviors they need to have.

There are many other diagrams we can use to model the system from different perspectives; interactions between objects, structure of the system, or the behavior of the system and how it responds to events. It's always about selecting the right diagram for the right need. You should realize which diagrams will be useful when thinking about or discussing a situation that isn't clear.

System Modeling

System modeling is the process of developing models of the system, with each model representing a different perspectives of that system. The most important aspect about a system model is that it leaves out detail; It's an abstract representation of the system. The models are usually based on graphical notation, which is almost always based on the notations in the Unified Modeling Language (UML). Other models of the system like mathematical model; a detailed system description.

Models are used during the analysis process to help to elicit the requirements, during the design process to describe the system to engineers, and after implementation to document the system structure and operation.

Notes

Different Perspectives

We may develop a model to represent the system from different perspectives. External, where you model the context or the environment of the system. Interaction, where you model the interaction between components of a system, or between a system and other systems.

Structural, where you model the organization of the system, or the structure of the data being processed by the system. Behavioral, where you model the dynamic behavior of the system and how it respond to events.

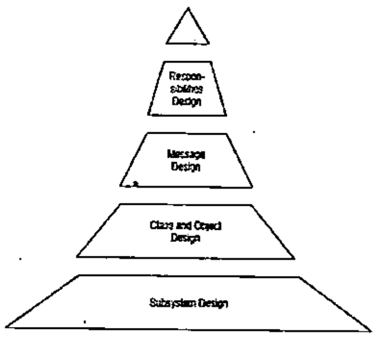
Unified Modeling Language (UML)

The unified modeling language become the standard modeling language for object-oriented modeling. It has many diagrams, however, the most diagrams that are commonly used are:

- Use case diagram: It shows the interaction between a system and it's environment (users or systems) within a particular situation.
- Class diagram: It shows the different objects, their relationship, their behaviors, and attributes.
- Sequence diagram: It shows the interactions between the different objects in the system, and between actors and the objects in a system.
- State machine diagram: It shows how the system respond to external and internal events.
- Activity diagram: It shows the flow of the data between the processes in the system. You can do diagramming work on paper or on a whiteboard, at least in the initial stages of a project. But there are some diagramming tools that will help you to draw these UML diagrams.

3.13 OBJECT ORIENTED DESIGN (OOD)

An analysis model created using object oriented analysis is transformed by object oriented design into a design model that works as a plan for software creation. OOD results in a design having several different levels of modularity i.e., The major system components are partitioned into subsystems (a system level "modular"), and data their manipulation operations are encapsulated into objects (a modular form that is the building block of an OO system.).



The Object Oriented Design Pyramid

In addition, OOD must specify some data organization of attributes and a procedural description of each operation.

Shows a design pyramid for object oriented systems. It is having the following four layers.

- The Subsystem Layer: It represents the subsystem that enables software to achieve user requirements and implement technical frameworks that meet user needs.
- The Class and Object Layer : It represents the class hierarchies that enable the system to develop using generalization and specialization. This layer also represents each object.
- The Message Layer : It represents the design details that enable each object to communicate with its partners. It establishes internal and external interfaces for the system.
- The Responsibilities Layer : It represents the data structure and algorithmic design for all the attributes and operations for each object.

The Object Oriented design pyramid specifically emphasizes specific product or system design. Note, however, that another design layer exists, which forms the base on which the pyramid rests. It focuses on the core layer the design of the domain object, which plays an important role in building the infrastructure for the Object Oriented system by providing support for human/computer interface activities, task management.

SUMMARY

- Analysis is the heart of the process. It is the key component of the first two phases of the cycle. In analysis the present system, the analyst collects a great deal of relatively unstructured data through interviews, questionnaires, on-site observations, procedures manuals, and the like.
- Requirements determination involves studying the current business system to find out how it works and where improvements should be made. Systems studies result in an evaluation of how current methods are working and whether adjustments are necessary or possible. These studies consider both manual and computermethods, they are not merely computer studies.
- The specific methods analysts use for collecting data about requirements are called fact – finding techniques. These include the interview, questionnaire, record inspections (on – site review) and observation. Analysts usually employ more that one of these techniques to help ensure an accurate and comprehensive investigation.
- Structured analysis is a set of techniques and graphical tools that allow the analyst to develop a new kind of system specifications that are easily understandable to the user. Analysts work primarily with their wits, pencil, and paper. Most of them have no tools. The traditional approach focuses on cost/benefit and feasibility analysis, project management, hardware and software selection and personnel considerations. In contrast, structured analysis considers new goals and structured tools for analysis.

Notes

- A DFD also known as a "bubble chart," has the purpose of clarifying system requirements and identifying major transformations that will be come programs in system design. So it is the starting point of the design phase that functionally decompose the requirements specifications down to the lowest level of detail.
- A DFD consists of a series of bubbles joined by lines. The bubbles represent data transformations and the lines represent data flows in the system. The system takes orders from the customer (bookstore, library, etc.), checks them against an index (file) listing the books available, verifies customer credit through a credit information file, and authorizes, shipment with an invoice.
- Entity Relationship Diagram is a data modeling method used in software engineering to produce a conceptual data model of an information system. Diagrams created using this ER-modeling method are called Entity-Relationship Diagrams or ER diagrams or ERDs.
- Data Dictionary is the major component in the structured analysis model of the system. It lists all the data items appearing in DFD. A data dictionary in Software Engineering means a file or a set of files that includes a database's metadata (hold records about other objects in the database), like data ownership, relationships of the data to another object, and some other data.
- Process analysis is the rational segmentation of business processes into different phases to understand them and look for ways to improve their overall efficiency. It refers to their full-fledged analysis, which also includes a series of logically articulated routine tasks that utilize organizational resources to transform a work process with the aim of benefiting the process and maintaining its excellence.
- Structured English is the use of the English language with the syntax of structured programming to communicate the design of a computer program to non-technical users by breaking it down into logical steps using straightforward English words. Structured English gives aims to get the benefits of both the programming logic and natural language: program logic helps to attain precision, whilst natural language helps with the familiarity of the spoken word
- A decision tree is a diagram that presents conditions and actions sequentially and thus shows which conditions to consider first, which second, and so on. It is also a method of showing the relationship of each condition and its permissible actions. The diagram resembles branches on a tree, hence the name.
- Object Oriented Analysis (OOA) is the first technical activity performed as part of object oriented software engineering. OOA is process of discovery where a development team understands and models the requirements of the system.
- In OOA requirements are organized as objects. It integrates all the process and data. But in others or traditional structural analysis both process and data are considered independently/separately. They use flow chart/structure charts for process and ER diagrams for data.

KEY WORDS

- **Process models:** They are usually set down through different graphing methods and are normally utilized to examine and represent a range of activities that occur periodically and on a regular basis. They can and are used to model the workflow between departments and people, or the flow of activities in an application or a system. These models have a set beginning and end, desired goal, structure of activities, and variable results based on the decisions taken through the process's course.
- **Business process modeling:** It is used to represent business processes graphically to recognize potential improvement or weak areas. Dealing specifically with low-level process maps, the main purpose of business process modeling is process improvement. Despite being extremely useful in concept, it is not typically used as a standalone procedure.
- **Data:** Representations of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automated means.
- **Data analysis:** (1) Evaluation of the description and intended use of each data item in the software design to ensure the structure and intended use will not result in a hazard. Data structures are assessed for data dependencies that circumvent isolation, partitioning, data aliasing, and fault containment issues affecting safety, and the control or mitigation of hazards. (2) Evaluation of the data structure and usage in the code to ensure each is defined and used properly by the program. Usually performed in conjunction with logic analysis.
- **Data flow analysis:** A software V&V task to ensure that the input and output data and their formats are properly defined, and that the data flows are correct.
- **Data flow diagra:** A diagram that depicts data sources, data sinks, data storage, and processes performed on data as nodes, and logical flow of data as links between the nodes. Syn: data flowchart, data flow graph.
- **Decision table:** A table used to show sets of conditions and the actions resulting from them.
- **Design:** The process of defining the architecture, components, interfaces, and other characteristics of a system or component. See: architectural design, preliminary design, detailed design.
- **Entity relationship diagram:** A diagram that depicts a set of real-world entities and the logical relationships among them.
- **Modeling:** Construction of programs used to model the effects of a postulated environment for investigating the dimensions of a problem for the effects of algorithmic processes on responsive targets.

Notes

REVIEW QUESTIONS

1. What is systems requirement.
3. What advantages do decision trees present from analysts.
4. Discuss the pros and cons of the various tools of doing analysis.
5. What is structured analysis.
6. Explain the entity relationship diagram.
7. What do you mean by data dictionary? Discuss.
8. What is decision tree? Explain.
9. Distinguish between OOA and OOD.
10. Discuss the process modelling.

FURTHER READINGS

1. Software Engineering, Bharat Bhushan Agarwal, Sumit Prakash Tayal, Firewall Media.
2. Softwaree Engineering, D. Sunder, University Science Press
3. Kaniclides, A., C. Kimble.(1995) A Development Framework for Executive Information Systems. Proceedings of GRONICS'95, Groningen, The Netherlands, February.
4. Kimble, C., (1997) Assessing the Relative Importance of Factors Affecting Information Systems Success, University of York Technical Report Series, YCS 283, Department of Computer Science, York, UK. 5.

Unit 4

Basics of Information Security

Notes

Structure

Notes

4.0 LEARNING OBJECTIVES

After reading this chapter students will be able to:

- define and describe tbasic information security
- discuss the types of attacks, virus controls, hackers
- understand an overview of risks associated with internet, internet security standards

4.1 INTRODUCTION

As of January 2008, the internet connected an estimated 541.7 million computers in more than 250 countries on every continent, even Antarctica The internet is not a single network, but a worldwide collection of loosely connected networks that are accessible by individual computer hosts, in a variety of ways, to anyone with a computer and a network connection. Thus, individuals and organizations can reach any point on the internet without regard to national or geographic boundaries or time of day.

However, along with the convenience and easy access to information come risks. Among them are the risks that valuable information will be lost, stolen, changed, or misused. If information is recorded electronically and is available on networked computers, it is more vulnerable than if the same information is printed on paper and locked in a file cabinet. Intruders do not need to enter an office or home; they may not even be in the same country. They can steal or tamper with information without touching a piece of paper or a photocopier. They can also create new electronic files, run their own programs, and hide evidence of their unauthorized activity.

4.2 BASIC SECURITY CONCEPTS

Three basic security concepts important to information on the internet are confidentiality, integrity, and availability. Concepts relating to the people who use that information are authentication, authorization, and nonrepudiation. When information is read or copied by someone not authorized to do so, the result is known as loss of confidentiality. For some types of information, confidentiality is a very important attribute. Examples include research data, medical and insurance records, new product specifications, and corporate investment strategies. In some locations, there may be a legal obligation to protect the privacy of individuals. This is particularly true for banks and loan companies; debt collectors; businesses that extend credit to their customers or issue credit cards; hospitals, doctors' offices, and medical testing laboratories; individuals or agencies that offer services such as psychological counseling or drug treatment; and agencies that collect taxes.

Information can be corrupted when it is available on an insecure network. When information is modified in unexpected ways, the result is known as loss of integrity. This means that unauthorized changes are made to information, whether by human error or intentional tampering. Integrity is particularly important for critical safety and financial data used for activities such as electronic funds transfers, air traffic control, and financial accounting. Information can be erased or become inaccessible, resulting in loss of availability. This means that people who are authorized to get information cannot get what they need. Availability is often the most important attribute in service-oriented businesses that depend on information (for example, airline schedules and online inventory systems).

Availability of the network itself is important to anyone whose business or education relies on a network connection. When users cannot access the network or specific services provided on the network, they experience a denial of service. To make information available to those who need it and who can be trusted with it, organizations use authentication and authorization. Authentication is proving that a user is the person he or she claims to be. That proof may involve something the user knows (such as a password), something the user has (such as a "smartcard"), or something about the user that proves the person's identity (such as a fingerprint). Authorization is the act of determining whether a particular user (or computer system) has the right to carry out a certain activity, such as reading a file or running a program.

Authentication and authorization go hand in hand. Users must be authenticated before carrying out the activity they are authorized to perform. Security is strong when the means of authentication cannot later be refuted—the user cannot later deny that he or she performed the activity. This is known as nonrepudiation.

These concepts of information security also apply to the term information security; that is, internet users want to be assured that

- they can trust the information they use
- the information they are responsible for will be shared only in the manner that they expect
- the information will be available when they need it
- the systems they use will process information in a timely and trustworthy manner

In addition, information assurance extends to systems of all kinds, including large-scale distributed systems, control systems, and embedded systems, and it encompasses systems with hardware, software, and human components. The technologies of information assurance address system intrusions and compromises to information.

What Can Happen

It is remarkably easy to gain unauthorized access to information in an insecure networked environment, and it is hard to catch the intruders. Even if users have nothing stored on their computer that they consider important, that computer can be a "weak link," allowing unauthorized access to the organization's systems and information.

Seemingly innocuous information can expose a computer system to compromise. Information that intruders find useful includes which hardware and software are being used, system configuration, type of network connections, phone numbers, and access and authentication procedures. Security-related information can enable unauthorized individuals to

Notes

access important files and programs, thus compromising the security of the system. Examples of important information are passwords, access control files and keys, personnel information, and encryption algorithms.

No one on the internet is immune. Those affected include banks and financial companies, insurance companies, brokerage houses, consultants, government contractors, government agencies, hospitals and medical laboratories, network service providers, utility companies, the textile business, universities, and wholesale and retail trades.

The consequences of a break-in cover a broad range of possibilities: a minor loss of time in recovering from the problem, a decrease in productivity, a significant loss of money or staff-hours, a devastating loss of credibility or market opportunity, a business no longer able to compete, legal liability, and the loss of life. Individuals may find that their credit card, medical, and other private information has been compromised. Identity theft can affect anyone.

Individuals who want to know more should read US-CERT Cyber Security Tips and other US-CERT papers. The US-CERT website contains papers, alerts, and other information for technical readers and for those responsible for government and control systems.

4.3 BASICS OF INFORMATION SECURITY

Information security is vital in an era in which data regarding countless individuals and organizations is stored in a variety of computer systems, often not under direct control. It is important to remember that security and productivity are often diametrically opposing concepts, and that being able to point out exactly when people are secure is a difficult task.

This unit covers some of the most basic concepts of information security. It discusses the diametrically opposing concepts of security and productivity, models that are helpful in discussing security concepts, such as the confidentiality, integrity, and availability (CIA) triad and the Parkerian hexad, as well as the basic concepts of risk and controls to mitigate it. Lastly, the chapter also covers defense in depth and its place in the information security world. Defense in depth is a particularly important concept in the world of information security.

To build defensive measures using this concept, multiple layers of defense are put in place, each giving an additional layer of protection. The idea behind defense in depth is not to keep an attacker out permanently but to delay him long enough to alert one to the attack and to allow one to mount a more active defense.

Definition

Various definitions of information security are suggested below, summarized from different sources:

- "Preservation of confidentiality, integrity and availability of information. Note: In addition, other properties, such as authenticity, accountability, non-repudiation and reliability can also be involved."
- "The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confiden-

tiality, integrity, and availability."

- "Ensures that only authorized users (confidentiality) have access to accurate and complete information (integrity) when required (availability)."
- "Information Security is the process of protecting the intellectual property of an organisation."
- "...information security is a risk management discipline, whose job is to manage the cost of information risk to the business."
- "A well-informed sense of assurance that information risks and controls are in balance."
- "Information security is the protection of information and minimizes the risk of exposing information to unauthorized parties."]

"Information Security is a multidisciplinary area of study and professional activity which is concerned with the development and implementation of security mechanisms of all available types (technical, organizational, human-oriented and legal) in order to keep information in all its locations (within and outside the organization's perimeter) and, consequently, information systems, where information is created, processed, stored, transmitted and destroyed, free from threats.

Threats to information and information systems may be categorized and a corresponding security goal may be defined for each category of threats. A set of security goals, identified as a result of a threat analysis, should be revised periodically to ensure its adequacy and conformance with the evolving environment.

The currently relevant set of security goals may include: confidentiality, integrity, availability, privacy, authenticity & trustworthiness, non-repudiation, accountability and auditability."

Information and information resource security using telecommunication system or devices means protecting information, information systems or books from unauthorized access, damage, theft, or destruction.

Overview

At the core of information security is information assurance, the act of maintaining the confidentiality, integrity, and availability (CIA) of information, ensuring that information is not compromised in any way when critical issues arise. These issues include but are not limited to natural disasters, computer/server malfunction, and physical theft.

While paper-based business operations are still prevalent, requiring their own set of information security practices, enterprise digital initiatives are increasingly being emphasized, with information assurance now typically being dealt with by information technology (IT) security specialists. These specialists apply information security to technology (most often some form of computer system). It is worthwhile to note that a computer does not necessarily mean a home desktop. A computer is any device with a processor and some memory. Such devices can range from non-networked standalone devices as simple as calculators, to networked mobile computing devices such as smartphones and tablet computers.

IT security specialists are almost always found in any major enterprise/establishment due to the nature and value of the data within larger businesses. They are responsible for keeping all of the technology within the company secure from malicious cyber attacks that often attempt to acquire critical private information or gain control of the internal systems.

The field of information security has grown and evolved significantly in recent years. It offers many areas for specialization, including securing networks and allied infrastructure, securing applications and databases, security testing, information systems auditing, business continuity planning, electronic record discovery, and digital forensics.

Information security professionals are very stable in their employment. As of 2013 more than 80 percent of professionals had no change in employer or employment over a period of a year, and the number of professionals is projected to continuously grow more than 11 percent annually from 2014 to 2019.

Threats

Information security threats come in many different forms. Some of the most common threats today are software attacks, theft of intellectual property, theft of identity, theft of equipment or information, sabotage, and information extortion. Most people have experienced software attacks of some sort. Viruses, worms, phishing attacks, and Trojan horses are a few common examples of software attacks.

The theft of intellectual property has also been an extensive issue for many businesses in the information technology (IT) field. Identity theft is the attempt to act as someone else usually to obtain that person's personal information or to take advantage of their access to vital information through social engineering.

Theft of equipment or information is becoming more prevalent today due to the fact that most devices today are mobile, are prone to theft and have also become far more desirable as the amount of data capacity increases. Sabotage usually consists of the destruction of an organization's website in an attempt to cause loss of confidence on the part of its customers. Information extortion consists of theft of a company's property or information as an attempt to receive a payment in exchange for returning the information or property back to its owner, as with ransomware.

There are many ways to help protect yourself from some of these attacks but one of the most functional precautions is conduct periodical user awareness. The number one threat to any organisation are users or internal employees, they are also called insider threats.

Governments, military, corporations, financial institutions, hospitals, non-profit organisations, and private businesses amass a great deal of confidential information about their employees, customers, products, research, and financial status. Should confidential information about a business' customers or finances or new product line fall into the hands of a competitor or a black hat hacker, a business and its customers could suffer widespread, irreparable financial loss, as well as damage to the company's reputation. From a business perspective, information security must be balanced against cost; the Gordon-Loeb Model provides a mathematical economic approach for addressing this concern.

For the individual, information security has a significant effect on privacy, which is viewed very differently in various cultures.

Responses to threats

Possible responses to a security threat or risk are:

- reduce/mitigate – implement safeguards and countermeasures to eliminate vulnerabilities or block threats
- assign/transfer – place the cost of the threat onto another entity or organization such as purchasing insurance or outsourcing
- accept – evaluate if the cost of the countermeasure outweighs the possible cost of loss due to the threat

4.4 TYPES OF ATTACKS

A security attack is an unauthorized attempt to steal, damage, or expose data from an information system such as your website. Malicious hackers can go about this in a variety of ways, including the ones listed below.

1. Malware

Malicious software – 'malware' – infects devices without users realizing it's there. Variations include Trojan horses, spyware, ransomware, 'malvertising', and viruses.

Secretly infected files or software can further introduce malware to your site. You could also trigger a malware download by clicking on a link in a pop-up window or an email attachment.

To prevent malware infections, you'll want to install a security scanner. This tool will alert you to otherwise undetected problems on your site. Our own security scanning feature, powered by Sucuri, is a low-cost and highly effective choice:

Phishing

Phishing is the practice of sending fraudulent communications that appear to come from a reputable source, usually through email. The goal is to steal sensitive data like credit card and login information or to install malware on the victim's machine. Phishing is an increasingly common cyberthreat.

Man-in-the-middle attack

Man-in-the-middle (MitM) attacks, also known as eavesdropping attacks, occur when attackers insert themselves into a two-party transaction. Once the attackers interrupt the traffic, they can filter and steal data.

Notes

Two common points of entry for MitM attacks:

1. On unsecure public Wi-Fi, attackers can insert themselves between a visitor's device and the network. Without knowing, the visitor passes all information through the attacker.
2. Once malware has breached a device, an attacker can install software to process all of the victim's information.

Denial-of-service attack

A denial-of-service attack floods systems, servers, or networks with traffic to exhaust resources and bandwidth. As a result, the system is unable to fulfill legitimate requests. Attackers can also use multiple compromised devices to launch this attack. This is known as a distributed-denial-of-service (DDoS) attack.

SQL injection

A Structured Query Language (SQL) injection occurs when an attacker inserts malicious code into a server that uses SQL and forces the server to reveal information it normally would not. An attacker could carry out a SQL injection simply by submitting malicious code into a vulnerable website search box.

Learn how to defend against SQL injection attacks.

Zero-day exploit

A zero-day exploit hits after a network vulnerability is announced but before a patch or solution is implemented. Attackers target the disclosed vulnerability during this window of time. Zero-day vulnerability threat detection requires constant awareness.

DNS Tunneling

DNS tunneling utilizes the DNS protocol to communicate non-DNS traffic over port 53. It sends HTTP and other protocol traffic over DNS. There are various, legitimate reasons to utilize DNS tunneling.

However, there are also malicious reasons to use DNS Tunneling VPN services. They can be used to disguise outbound traffic as DNS, concealing data that is typically shared through an internet connection. For malicious use, DNS requests are manipulated to exfiltrate data from a compromised system to the attacker's infrastructure. It can also be used for command and control callbacks from the attacker's infrastructure to a compromised system.

4.5 VIRUSES

A virus is a computer code or program, which is capable of affecting your computer data badly by corrupting or destroying them.

Notes

Computer virus has the tendency to make its duplicate copies at a swift pace, and also spread it across every folder and damage the data of your computer system.

A computer virus is actually a malicious software program or "malware" that, when infecting your system, replicates itself by modifying other computer programs and inserting its own code.

Fig. 1. Infected computer programs may include data files, or even the "boot" sector of the hard drive.

Types of Virus

Following are the major types of computer virus –

- **Worms:** This is a computer program that replicates itself at a swift pace. Unlike a computer virus, it is self-contained and hence does not need to be part of another program to propagate itself. A worm is computer code that spreads without user interaction. Most worms begin as email attachments that infect a computer when they're opened. The worm scans the infected computer for files, such as address books or temporary webpages, that contain email addresses. The worm uses the addresses to send infected email messages, and frequently mimics (or spoofs) the "From" addresses in later email messages so that those infected messages seem to be from someone you know. Worms then spread automatically through email messages, networks, or operating system vulnerabilities, frequently overwhelming those systems before the cause is known. Worms aren't always destructive to computers, but they usually cause computer and network performance and stability problems.
- **Trojan Horse:** A trojan horse is a malicious software program that hides inside other programs. It enters a computer hidden inside a legitimate program, such as a screen saver. Then it puts code into the operating system that enables a hacker to access the infected computer. Trojan horses do not usually spread by themselves. They are spread by viruses, worms, or downloaded software.
- **Bombs:** It is similar to Trojan Horse, but Logic bombs have some specialty; these include a timing device and hence it will go off only at a particular date and time.

Notes

How Does Virus Affect?

Let us discuss in what ways a virus can affect your computer system. The ways are mentioned below –

- By downloading files from the Internet.
- During the removable of media or drives.
- Through pen drive.
- Through e-mail attachments.
- Through unpatched software & services.
- Through unprotected or poor administrator passwords.

Impact of Virus

Let us now see the impact of virus on your computer system –

- Disrupts the normal functionality of respective computer system.
- Disrupts system network use.
- Modifies configuration setting of the system.
- Destructs data.
- Disrupts computer network resources.
- Destructs of confidential data.

Virus Detection

The most fundamental method of detection of virus is to check the functionality of your computer system; a virus affected computer does not take command properly.

However, if there is antivirus software in your computer system, then it can easily check programs and files on a system for virus signatures.

4.6 VIRUS CONTROL

How are computer viruses removed?

Antiviruses have made great progress in being able to identify and prevent the spread of computer viruses. When a device does become infected, though, installing an antivirus solution is still your best bet for removing it. Once installed, most software will conduct a "scan" for the malicious program.

Once located, the antivirus will present options for its removal. If this is not something that can be done automatically, some security vendors offer a technician's assistance in removing the virus free of charge.

Examples of computer viruses

In 2013, the botnet virus Gameover ZueS was discovered to use peer-to-peer downloading sites to distribute ransomware and commit banking fraud. While tens of thousands of computer viruses still roam the internet, they have diversified their methods and are now joined by several malware variants like:

- Worms - A worm is a type of virus that, unlike traditional viruses, usually does not require the action of a user to spread from device to device.
- Trojans - As in the myth, a Trojan is a virus that hides within a legitimate-seeming program to spread itself across networks or devices.
- Ransomware - Ransomware is a type of malware that encrypts a user's files and demands a ransom for its return. Ransomware can be, but isn't necessarily, spread through computer viruses.

Computer virus protection

When you arm yourself with information and resources, you're wiser about computer security threats and less vulnerable to threat tactics. Take these steps to safeguard your PC with the best computer virus protection:

- Use antivirus protection and a firewall
- Get antispyware software
- Always keep your antivirus protection and antispyware software up-to-date
- Update your operating system regularly
- Increase your browser security settings
- Avoid questionable Websites
- Only download software from sites you trust.
- Carefully evaluate free software and file-sharing applications before downloading them.
- Don't open messages from unknown senders
- Immediately delete messages you suspect to be spam

An unprotected computer is like an open door for computer viruses. Firewalls monitor Internet traffic in and out of your computer and hide your PC from online scammers looking for easy targets.

Products like Webroot Internet Security Complete and Webroot Antivirus provide complete protection from the two most dangerous threats on the Internet - spyware and computer viruses. They prevent viruses from entering your computer, stand guard at every possible entrance of your computer and fend off any computer virus that tries to open, even the most damaging and devious strain.

Notes

Virus Preventive Measures

Let us now see the different virus preventive measures. A computer system can be protected from virus through the following –

- Installation of an effective antivirus software.
- Patching up the operating system.
- Patching up the client software.
- Putting highly secured Passwords.
- Use of Firewalls.
- Most Effective Antivirus

Following are the most popular and effective antivirus from which you can choose one for your personal computer –

- McAfee Antivirus Plus
- Symantec Norton Antivirus
- Avast Pro Antivirus
- Bitdefender Antivirus Plus
- Kaspersky Anti-Virus
- Avira Antivirus
- Webroot Secure Anywhere Antivirus
- Emsisoft Anti-Malware
- Quick Heal Antivirus
- ESET NOD32 Antivirus

4.7 HACKERS

Hacking is the activity of identifying weaknesses in a computer system or a network to exploit the security to gain access to personal data or business data. An example of computer hacking can be: using a password cracking algorithm to gain access to a computer system.

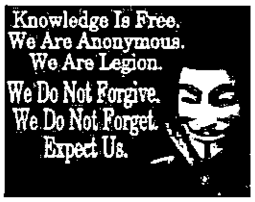
Computers have become mandatory to run a successful businesses. It is not enough to have isolated computers systems; they need to be networked to facilitate communication with external businesses. This exposes them to the outside world and hacking. System hacking means using computers to commit fraudulent acts such as fraud, privacy invasion, stealing corporate/personal data, etc. Cyber crimes cost many organizations millions of dollars every year. Businesses need to protect themselves against such attacks. Before we learn hacking, let's look at the introduction of hacking and some of the most commonly used terminologies in the world of hacking.

Who is a Hacker?

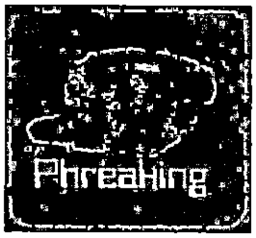
A Hacker is a person who finds and exploits the weakness in computer systems and/or networks to gain access. Hackers are usually skilled computer programmers with knowledge of computer security.

Types of Hackers

Hackers are classified according to the intent of their actions. The following list classifies types of hackers according to their intent:

Symbol	Description
	Ethical Hacker (White hat): A security hacker who gains access to systems with a view to fix the identified weaknesses. They may also perform penetration Testing and vulnerability assessments.
	Cracker (Black hat): A hacker who gains unauthorized access to computer systems for personal gain. The intent is usually to steal corporate data, violate privacy rights, transfer funds from bank accounts etc.
	Grey hat: A hacker who is in between ethical and black hat hackers. He/she breaks into computer systems without authority with a view to identify weaknesses and reveal them to the system owner.
	Script kiddies: A non-skilled person who gains access to computer systems using already made tools.
 Knowledge Is Free. We Are Anonymous. We Are Legion. We Do Not Forgive. We Do Not Forget. Expect Us.	Hacktivist: A hacker who use hacking to send social, religious, and political, etc. messages. This is usually done by hijacking websites and leaving the message on the hijacked website.

Notes

	Phreaker: A hacker who identifies and exploits weaknesses in telephones instead of computers.

Cybercrime

Cybercrime is the activity of using computers and networks to perform illegal activities like spreading computer viruses, online bullying, performing unauthorized electronic fund transfers, etc. Most cybercrime hacks are committed through the internet, and some cybercrimes are performed using Mobile phones via SMS and online chatting applications.

Type of Cybercrime

The following list presents the common types of cybercrimes:

- Computer Fraud: Intentional deception for personal gain via the use of computer systems.
- Privacy violation: Exposing personal information such as email addresses, phone number, account details, etc. on social media, hacking a websites, etc.
- Identity Theft: Stealing personal information from somebody and impersonating that person.
- Sharing copyrighted files/information: This involves distributing copyright protected files such as eBooks and computer programs etc.
- Electronic funds transfer: This involves gaining an un-authorized access to bank computer networks and making illegal fund transfers.
- Electronic money laundering: This involves the use of the computer to launder money.
- ATM Fraud: This involves intercepting ATM card details such as account number and PIN numbers. These details are then used to withdraw funds from the intercepted accounts.
- Denial of Service Attacks: This involves the use of computers in multiple locations to attack servers with a view of shutting them down.
- Spam: Sending unauthorized emails. These emails usually contain advertisements.

What is Ethical Hacking?

Ethical Hacking is identifying weakness in computer systems and/or computer networks and coming with countermeasures that protect the weaknesses. Ethical hackers must abide by the following rules.

- Get written permission from the owner of the computer system and/or computer network before hacking.
- Protect the privacy of the organization been hacked.
- Transparently report all the identified weaknesses in the computer system to the organization.
- Inform hardware and software vendors of the identified weaknesses.

Why Ethical Hacking?

- Information is one of the most valuable assets of an organization. Keeping information secure can protect an organization's image and save an organization a lot of money.
- Fake hacking can lead to loss of business for organizations that deal in finance such as PayPal. Ethical hacking puts them a step ahead of the cyber criminals who would otherwise lead to loss of business.

Legality of Ethical Hacking

Ethical Hacking is legal if the hacker abides by the rules stipulated in the above section on the definition of ethical hacking. The International Council of E-Commerce Consultants (EC-Council) provides a certification program that tests individual's skills. Those who pass the examination are awarded with certificates. The certificates are supposed to be renewed after some time.

In conclusion we can say hacking is identifying and exploiting weaknesses in computer systems and/or computer networks. Cybercrime is committing a crime with the aid of computers and information technology infrastructure. Ethical Hacking is about improving the security of computer systems and/or computer networks. Ethical Hacking is legal.

4.8 OVERVIEW OF RISKS ASSOCIATED WITH INTERNET

The advance of digital media has created risks that affect the bio-psycho-social well-being of adolescents. Some of these risks are cyberbullying, cyber dating abuse, sexting, online grooming and problematic Internet use. These risks have been studied individually or through associations of some of them but they have not been explored conjointly. The main objective is to determine the comorbidity between the described Internet risks and to identify the profiles of victimized adolescents.

The digital society is an opportunity for personal development in many fields related to social, health, educational and economic aspects. Although digital media provides great advantages (such as rapid communication, information availability, opportunities for learning and entertainment), its use is not without potential risks.

According to the co-construction model, through digital media, adolescents construct and co-construct their environments, connecting their online and offline worlds. Therefore, digital worlds may serve as a playground for important developmental tasks of adolescence such as sexuality and identity but also, adolescents have to deal with the darker and more

Notes

unsavoury aspects of technology], during a stage of special psychological vulnerability. Given that digital media presents some unique challenges for sexuality, intimacy, aggressive behaviour and problematic use that affect boys and girls and older and younger youth in different ways, participants' details (such as their sex and age) are an important issue when studying online behaviour.

High-risk situations are mainly due to the recent substantial increase in the use of Internet. This can be confirmed through the survey published in 2015 by the Pew Research Centre, which reported that 24% of youngsters between 13 and 17 years are constantly connected to Internet and 56% of them several times a day.

In this same line, the results of the CIBERASTUR (Project about bullying, cyberbullying, problematic uses of internet and quality of life related to health between 10 and 18 years of the Principality of Asturias) study performed in Spain with more than 25,000 adolescents aged 11–18 years showed that 95.7% reported owning a smartphone and, among them, up to 86.6% used it daily. Regarding the time of connection, 20.6% of the adolescents reported spending five hours or more from Monday to Friday and on weekends, this increased to 33.2%.

The increasing possession of smartphones, as well as their greater everyday use, is the gateway to potential risks that negatively affect bio-psycho-social well-being.

In this unit, there are many approaches under the heading of Internet risks. The authors of this study define these risks as a set of psychosocial problems that are characteristic of Internet, initiated and maintained in an online context that has a mutual and bidirectional relationship with the individual's off-line reality. These risks can have severe outcomes for victims, who often present internalizing and externalizing problem, loss of perceived quality of life, suicidal ideation and interference in academic, social and family life.

Moreover, although most of these risks have been assessed, particularly cyberbullying, the interconnections among them have been neglected. However, research has shown that victimization in one context can make youth vulnerable to other types of victimization and thus extend their victim status over time.

According to the polyvictimization theory, victimization often does not occur in isolation but is frequently followed by other forms of abuse. These authors highlighted the importance of identifying children and adolescents who had experienced multiple victimizations because they could develop more severe psychological problems.

In this sense, research shows that populations involved in multiple risk behaviours have the greatest risk for chronic diseases, psychiatric disorders, suicidal behaviours and premature death compared to individuals with single or no risk behaviours. This also falls in line with several theories, such as the cumulative risk model, which states that most children experiencing a single psychosocial risk factor might suffer little or no enduring harm, whereas a subset of children experiencing multiple risk factors are much more likely to experience psychological disorders.

In line with the above-mentioned theories, this study aims to explore five possible adolescent risks in the digital media (cyberbullying victimization, cyber dating abuse victimization, sexting, online grooming and problematic Internet use), considering their interconnections and comorbidities.

Cyberbullying is a violent and intentional act that is performed repeatedly, over a long period of time, through the use of the new technologies, by one or more persons directed against another person who has difficulties to defend him- or herself. In addition, it is usually anonymous and can occur at any time and place.

To form a general idea of the current situation, a review of 159 studies determined that the prevalence of cyberbullying last year ranged between 1% and 61.1%. In this sense, a study on cyberbullying in all the Spanish regions showed average victimization values near 25%.

The prevalence of victimization tends to be higher in girls than in boys, although some reviews indicate that the results are mixed and there is no unanimity. Regarding differences based on age, cyberbullying appears to increase as children approach adolescence, reaching its maximum prevalence around age 15.

In comparison with cyberbullying, cyber dating abuse has received less attention. This phenomenon comprises a wide range of behaviours that include, for example, attempts to control one's partner or ex-partner via digital media and/or by sending insulting or threatening messages. Regarding victimization in Spain, rates of 75% and 14%, respectively, for controlling behaviour and insulting or threatening messages, have been obtained.

In terms of sex differences, although girls report higher sexual victimization than boys between ages 12-18, the results depend on the type of behaviour analysed and are disparate in age ranges above 18 years.

Online grooming is a serious social problem and it is often considered a criminal offence, like in Spain, where it is included in the Criminal Code. Grooming has been defined as the process by which an adult, using digital media prepares a minor in order to obtain sexual material (images, videos) from him or her or to sexually abuse him or her.

Studies of surveys of adolescents aged 10-17 indicate a prevalence of sexual requests ranging between 5% and 9%. Prevalence figures of 15.6% for girls and 9.3% for boys have been found in a Spanish population study of young people aged 12 to 15 years.

Sexting refers to the act of sending a peer photographs and videos with some level of sexual content, taken or recorded by the protagonist, through the use of digital media. In the international context, the prevalence data on sexting vary between 9.6% and 54%. In Spain, the few works carried out found a sexting prevalence rate of 13.5%. Although some studies suggest that sexting is practiced more by girls, other studies have revealed that there seems to be no differences between boys and girls, with prevalence rising as age increases.

Finally, problematic Internet use stresses the possible dysfunctions that Internet consumption can imply in the person's life. Preference for online social interaction and mood regulation through Internet increase the likelihood of presenting poor self-regulation, which has several negative consequences in the person's life.

In a recent study in Spain, 4% of the students presented problematic use or a pattern of risk and nearly 40% had presented some occasional problematic use in the last seven months. The epidemiological studies report prevalence rates that reveal the clinical and social relevance of this problem, which, moreover, is increasing with age. In relation to sex differences, greater problematic Internet use has been observed in females although other studies find no differences.

Notes

As mentioned, the available research has generally focused on a single risk or has only addressed the associations between some of these phenomena. Thus, recent studies have linked cyberbullying with problematic Internet use, cyber dating abuse victimization with problematic Internet use and sexting and cyber dating abuse victimization and perpetration with cyberbullying victimization and aggression, sexting with cyber dating abuse perpetration and victimization and grooming with sexting and cyberbullying victimization.

However, to our knowledge, the comorbidity of this series of risks has not been evaluated in order to establish possible typologies of risks. The development of typologies that integrate several modalities of Internet victimization could contribute to extend the polyvictimization theory.

Therefore, the main objective of this work is to determine the comorbidity between Internet risks (cyberbullying victimization, cyber dating abuse victimization, sexting, online grooming and problematic Internet use) and to identify the profiles of adolescents as a function of the presence of these risks. The secondary goals are to provide the latest data on the prevalence of victimization due to Internet risks and to analyse differences according to sex, educational stage and type of school (private or public centre).

We expect to find high comorbidity among the risks and to observe the emergence of several distinct profiles. Regarding the secondary objectives, considering the results of previous studies, our working hypothesis is that cyberbullying victimization will be the Internet risk with the highest prevalence. Like in diverse studies, we expect to find differences between boys and girls, with higher victimization scores in girls. We also expect that the higher the educational stage, the higher will be the scores in the diverse risks, given that age facilitates access to Internet.

Cyberbullying

Bullying implies an intention to harm, intimidate or coerce an act when there is an imbalance of power and the act is a cause for distress and provocation. Bullying may be verbal, physical or mental in nature and a whole spectrum of acts can constitute bullying.

It can become a source of trauma for children and young adults and remain with them their whole lives, often leading to mental distress and depression, and in extreme situations, even suicide.

What is central to bullying is an imbalance of power dynamics. Bullies are usually physically stronger than the people they bully. It can be described as a show of strength to undermine or denigrate someone's dignity to gain sadistic pleasure out of it.

Bullying is a common phenomenon in schools and universities throughout the country and there has been a concerted effort to put an end to bullying and ragging in our educational institutions.

Online predator

Online predators are individuals who commit child sexual abuse that begins or takes place on the Internet. Internet-facilitated crimes against minors involve deceit and begin with adults communicating with children over the Internet with the goal of coercing them into illegal sexual activity. Sometimes the sexual abuse happens face to face.

Chat rooms, instant messaging, Internet forums, social networking sites, cell phones, and even video game consoles have issues with online predations. These online areas attract predators because they allow them to have access to make contact with victims without drawing attention.

In addition, there is insufficient reliable data concerning the number of minors sharing personal information online due to children's privacy issues. Also, the anonymity of online conversations leads to the disinhibition of minors, making them feel more comfortable and more likely to engage in risky behaviors.

This allows predators to use manipulation to put their targets into situations where they will comply with the predator's sexual demands. Initial manipulation often involves introducing the minors to sexual activity, showing them pornography, and requesting sexually explicit information and pictures. This online predatory behavior does not often lead to actual or attempted offline contact, but it could.

Even though it is the mainstream view that predators will use distinct tactics to meet victims, most actual in-person meetings do not involve any deception. In fact, the minors are usually complicit with perpetrators often using promises of love and romance to seduce victims to meet

Theft of Personal Information

Everything that you post on social networks is permanent and un like a letter it can't be torn or burned .Today's youth don't understand the damage a random picture or post they u ploaded on social networks can affect their life in future .Also if you r social account is hacked , the hacker can know all your personal information and harass you. If your bank or credit card details get leaked, you can suffer extensive damage.

Identity theft occurs when someone uses another person's personal identifying information, like their name, identifying number, or credit card number, without their permission, to commit fraud or other crimes. The term identity theft was coined in 1964. Since that time, the definition of identity theft has been statutorily defined throughout both the U.K. and the United States as the theft of personally identifiable information.

Identity theft deliberately uses someone else's identity as a method to gain financial advantages or obtain credit and other benefits, and perhaps to cause other person's disadvantages or loss. The person whose identity has been stolen may suffer adverse consequences, especially if they are falsely held responsible for the perpetrator's actions.

Personally identifiable information generally includes a person's name, date of birth, social security number, driver's license number, bank account or credit card numbers, PINs, electronic signatures, fingerprints, passwords, or any other information that can be used to access a person's financial resources.

Determining the link between data breaches and identity theft is challenging, primarily because identity theft victims often do not know how their personal information was obtained. According to a report done for the FTC, identity theft is not always detectable by the individual

Notes

victims. Identity fraud is often but not necessarily the consequence of identity theft. Someone can steal or misappropriate personal information without then committing identity theft using the information about every person, such as when a major data breach occurs.

A US Government Accountability Office study determined that "most breaches have not resulted in detected incidents of identity theft". The report also warned that "the full extent is unknown". A later unpublished study by Carnegie Mellon University noted that "Most often, the causes of identity theft is not known", but reported that someone else concluded that "the probability of becoming a victim to identity theft as a result of a data breach is ... around only 2%".

For example, in one of the largest data breaches which affected over four million records, it resulted in only about 1,800 instances of identity theft, according to the company whose systems were breached.

An October 2010 article entitled "Cyber Crime Made Easy" explained the level to which hackers are using malicious software. As Gunter Ollmann, Chief Technology Officer of security at Microsoft, said, "Interested in credit card theft? There's an app for that." This statement summed up the ease with which these hackers are accessing all kinds of information online. The new program for infecting users' computers was called Zeus, and the program is so hacker-friendly that even an inexperienced hacker can operate it.

Although the hacking program is easy to use, that fact does not diminish the devastating effects that Zeus (or other software like Zeus) can do on a computer and the user. For example, programs like Zeus can steal credit card information, important documents, and even documents necessary for homeland security. If a hacker were to gain this information, it would mean identity theft or even a possible terrorist attack. The ITAC says that about 15 million Americans had their identity stolen in 2012.

Pornography and other inappropriate content

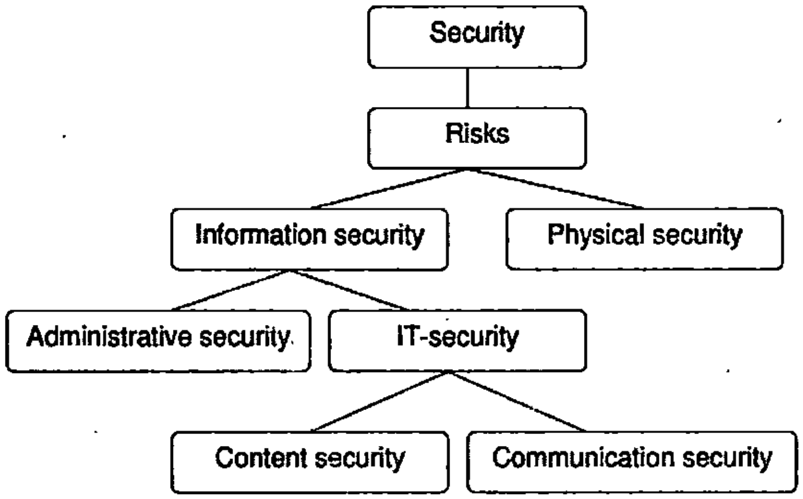
Internet is filled with adult and inappropriate content that can steal the innocence and morality of your chi ldren.You can find links to adult websites on social med ia and other sites that your children may like to visit. An accidental cl ick on these links can take them on a trip to the filthy and grotesque world of pornography.

4.9 INTRUSION DETECTION RISK MANAGEMENT

There are different ways to approach information security. Very simplified, there are two basic approaches. A static approach that relies heavily on user rules and regulations and static IT-environments that does not change often and therefore can be scrutinously analyzed and protected, usually by border protection. The static approach, again simplified, regards an asset in need of protection as binary; either secure or insecure.

Another, perhaps more modern approach is to utilize a dynamic approach based onrisk management. This approach does not regard assets in a binary manner, instead tries to analyze the asset and its environment in its full diversity of available security threats and protection mechanisms. The real world of course uses a combination of the two approaches, usually where the first approach is used as a security baseline for an organization and the second approach is used to evaluate if further protection is needed (that is more protection than that of what the baseline is offering) and if so what kind of protection.

Risk management is a popular tool for today's information security professionals and is therefore frequently put to practice. There are many different methodologies and standards available on the market today. In order to unify all the available standards and methods offered today there exist an international standardization effort regarding risk management in an ISO/IEC project, conducted by ISO/TMB/WG Risk Management.



4.10 RISK MANAGEMENT

A software project can be concerned with a large variety of risks. In order to be adept to systematically identify the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks which can affect a software project:

- Project risks
- Technical risks
- Business risks

1. **Project risks:** Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project. It is very tough to control something which cannot be identified. For any manufacturing program, such as the manufacturing of cars, the plan executive can recognize the product taking shape.

Notes

2. **Technical risks:** Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.
3. **Business risks:** This type of risks contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

Other risk categories

1. Known risks: Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)
2. Predictable risks: Those risks that are hypothesized from previous project experience (e.g., past turnover)
3. Unpredictable risks: Those risks that can and do occur, but are extremely tough to identify in advance.

Principle of Risk Management

- **Global Perspective:** In this, we review the bigger system description, design, and implementation. We look at the chance and the impact the risk is going to have.
- **Take a forward-looking view:** Consider the threat which may appear in the future and create future plans for directing the next events.
- **Open Communication:** This is to allow the free flow of communications between the client and the team members so that they have certainty about the risks.
- **Integrated management:** In this method risk management is made an integral part of project management.
- **Continuous process:** In this phase, the risks are tracked continuously throughout the risk management paradigm.

Broadly speaking, risk is the likelihood that something bad will happen that causes harm to an informational asset (or the loss of the asset). A vulnerability is a weakness that could be used to endanger or cause harm to an informational asset.

A threat is anything (man-made or act of nature) that has the potential to cause harm. The likelihood that a threat will use a vulnerability to cause harm creates a risk. When a threat does use a vulnerability to inflict harm, it has an impact. In the context of information security, the impact is a loss of availability, integrity, and confidentiality, and possibly other losses (lost income, loss of life, loss of real property).

The Certified Information Systems Auditor (CISA) Review Manual 2006 defines risk management as "the process of identifying vulnerabilities and threats to the information resources used by an organization in achieving business objectives, and deciding what countermeasures, if any, to take in reducing risk.to an acceptable level, based on the value of the information resource to the organization."

There are two things in this definition that may need some clarification.

- First, the process of risk management is an ongoing, iterative process. It must be repeated indefinitely. The business environment is constantly changing and new threats and vulnerabilities emerge every day.
- Second. the choice of countermeasures (controls) used to manage risks must strike a balance between productivity, cost, effectiveness of the countermeasure, and the value of the informational asset being protected.

Furthermore, these processes have limitations as security breaches are generally rare and emerge in a specific context which may not be easily duplicated. Thus, any process and countermeasure should itself be evaluated for vulnerabilities. It is not possible to identify all risks, nor is it possible to eliminate all risk. The remaining risk is called "residual risk."

A risk assessment is carried out by a team of people who have knowledge of specific areas of the business. Membership of the team may vary over time as different parts of the business are assessed. The assessment may use a subjective qualitative analysis based on informed opinion, or where reliable dollar figures and historical information is available, the analysis may use quantitative analysis.

Research has shown that the most vulnerable point in most information systems is the human user, operator, designer, or other human. The ISO/IEC 27002:2005 Code of practice for information security management recommends the following be examined during a risk assessment:

- security policy,
- organization of information security,
- asset management,
- human resources security,
- physical and environmental security,
- communications and operations management,
- access control,
- information systems acquisition, development, and maintenance,
- information security incident management,
- business continuity management
- regulatory compliance.

In broad terms, the risk management process consists of:

- Identification of assets and estimating their value. Include: people, buildings, hardware, software, data (electronic, print, other), supplies.
- Conduct a threat assessment. Include: Acts of nature, acts of war, accidents, malicious acts originating from inside or outside the organization..
- Conduct a vulnerability assessment, and for each vulnerability, calculate the probability that it will be exploited. Evaluate policies, procedures, standards, training, physical

security, quality control, technical security.

- Calculate the impact that each threat would have on each asset. Use qualitative analysis or quantitative analysis.
- Identify, select and implement appropriate controls. Provide a proportional response. Consider productivity, cost effectiveness, and value of the asset.
- Evaluate the effectiveness of the control measures. Ensure the controls provide the required cost effective protection without discernible loss of productivity.

For any given risk, management can choose to accept the risk based upon the relative low value of the asset, the relative low frequency of occurrence, and the relative low impact on the business.

Or, leadership may choose to mitigate the risk by selecting and implementing appropriate control measures to reduce the risk. In some cases, the risk can be transferred to another business by buying insurance or outsourcing to another business. The reality of some risks may be disputed. In such cases leadership may choose to deny the risk.

Security controls

Selecting and implementing proper security controls will initially help an organization bring down risk to acceptable levels. Control selection should follow and should be based on the risk assessment. Controls can vary in nature, but fundamentally they are ways of protecting the confidentiality, integrity or availability of information.

ISO/IEC 27001 has defined controls in different areas. Organizations can implement additional controls according to requirement of the organization ISO/IEC 27002 offers a guideline for organizational information security standards.

Administrative Control

Administrative controls (also called procedural controls) consist of approved written policies, procedures, standards, and guidelines. Administrative controls form the framework for running the business and managing people. They inform people on how the business is to be run and how day-to-day operations are to be conducted. Laws and regulations created by government bodies are also a type of administrative control because they inform the business.

Some industry sectors have policies, procedures, standards, and guidelines that must be followed – the Payment Card Industry Data Security Standard (PCI DSS) required by Visa and MasterCard is such an example. Other examples of administrative controls include the corporate security policy, password policy, hiring policies, and disciplinary policies.

Administrative controls form the basis for the selection and implementation of logical and physical controls. Logical and physical controls are manifestations of administrative controls, which are of paramount importance.

Logical Control

Logical controls (also called technical controls) use software and data to monitor and control access to information and computing systems. Passwords, network and host-based firewalls, network intrusion detection systems, access control lists, and data encryption are examples of logical controls.

An important logical control that is frequently overlooked is the principle of least privilege, which requires that an individual, program or system process not be granted any more access privileges than are necessary to perform the task. A blatant example of the failure to adhere to the principle of least privilege is logging into Windows as user Administrator to read email and surf the web. Violations of this principle can also occur when an individual collects additional access privileges over time.

This happens when employees' job duties change, employees are promoted to a new position, or employees are transferred to another department. The access privileges required by their new duties are frequently added onto their already existing access privileges, which may no longer be necessary or appropriate.

Physical Control

Physical controls monitor and control the environment of the work place and computing facilities. They also monitor and control access to and from such facilities and include doors, locks, heating and air conditioning, smoke and fire alarms, fire suppression systems, cameras, barricades, fencing, security guards, cable locks, etc. Separating the network and workplace into functional areas are also physical controls.

An important physical control that is frequently overlooked is separation of duties, which ensures that an individual can not complete a critical task by himself. For example, an employee who submits a request for reimbursement should not also be able to authorize payment or print the check. An applications programmer should not also be the server administrator or the database administrator; these roles and responsibilities must be separated from one another.

Motivation for Risk Management

Traditional network security has a static approach towards defense. Risk Management on the other hand analyzes threats, evaluated and graded threats according to the principles stipulated by risk management.

Traditional security leads to static defense mechanisms such as firewalls and Virtual Private Networks (VPN) with static rules that clearly dictate which network traffic is allowed and which is not. This is regardless of the generation [CISSP] of the firewall technology as firewalls are based on the concept of clear boundaries between an inside of a network and an outside that contain the threats towards the network. In today's complex and integrated network world the concept of a clear boundary is no longer relevant. The defense mechanisms must therefore be concentrated elsewhere.

Modern networks are extremely dynamic. By this I mean that the networks aren't based on just one protocol nor does the network only allow single protocols to access the networks such as only HTTP traffic through a single access point. Instead the modern

4.11 DISASTER RECOVERY PLAN

Disaster recovery is all about making sure your business can continue operating with minimal losses in the event of a disaster.

Cybersecurity disaster recovery focuses explicitly on disasters resulting from cyber threats, such as DDoS attacks or data breaches.

Your recovery plan will detail the steps your organization needs to take to stop losses, end the threat, and move on without jeopardizing the future of the business. These are some of the biggest goals you'll need to achieve with any plan you develop.

1. Business Continuity

First and foremost, you need to establish a line of business continuity.

In other words, your highest priority needs to be making sure that the business can continue operating during and immediately after the threat. This way, you can continue generating revenue. In addition, you'll want to maintain your reputation as you pick up the pieces in the wake of the disaster.

2. Data protection.

You'll also need to think about protecting your data.

This includes minimizing data accessibility to hackers, reducing the threat of data loss, and making it possible to back up your data when the threat is over.

3. Loss minimization.

Businesses can suffer various other losses and forms of damage in the wake of a disaster.

These include financial losses, legal ramifications, and reputational blows. Therefore, part of your disaster recovery plan needs to focus on minimizing these losses.

4. Communication.

You also need to think about how you will communicate this disaster, both internally and externally.

How will you make sure all your staff members are up-to-date about what has happened? And how are you going to break the news to stakeholders?

5. Restoration.

Once the threat has been mitigated or completely ended, you can focus on restoration.

What steps do you need to take to restore your systems back to normal, and what's the fastest and most efficient path to do this?

6. Improvements.

Every disaster recovery plan should also have a phase documented for reflection and improvement.

Why did this threat jeopardize your business? What did you do right? What did you do wrong? And what improvements can you make in the future?

Choose the proper authorities.

Before you start sketching out your disaster recovery plan, it's a good idea to consider which authorities you want to trust on this subject.

Many businesses choose to outsource some of these responsibilities. Instead, they hire an IT support service provider to help them evaluate their potential risks and assemble a recovery plan.

Failing that, it's a good idea to designate one person in your organization to be in charge of signing off on the final plan and executing that plan in the event of a cyber security disaster. This could be your CTO, the head of your IT department, or some other authority.

Invest in prevention.

In a perfect world. you'll never need a disaster recovery plan because you'll never face a cybersecurity disaster. That's why it's a good idea to invest in prevention as much as you invest in recovery, if not more so.

- Firewalls and VPNs. Firewalls and VPNs give you more control over traffic and accessibility on your network.
- Updates and upgrades. Staying up to date with the latest software patches and best practices can help you guard against the majority of recently revealed vulnerabilities.
- Strict content controls. Internal content controls can prevent unauthorized access to your most important data and applications.
- Accessibility limitations. If a smaller number of people can access your company's most sensitive data, you'll bear fewer risks.
- Staff education. The majority of security exploits are a direct result of human error. As a result, it pays to train and educate your staff on best practices for cybersecurity.
- Identify your most significant potential threats.
- One of the most essential phases of your cybersecurity disaster recovery planning is identifying your most significant potential threats.

You'll need to identify the potential hacks, attacks, breaches, and exploits that could threaten your organization and understand the risks associated with those events.

It's also important to understand the consequences of those threats. For example, how will your finances be affected if you face one of these threats? What legal consequences could there be? How will stakeholders respond to such a threat?

Notes

Once you understand both the likelihood and the consequences of a given threat, you'll be able to contextualize it and understand its priority level.

Establish a monitoring plan.

How are you going to monitor for these threats? Well-prepared businesses have an ongoing monitoring program in place.

It allows them to notice when a breach is underway, or identify a threat before it's too late. Consequently, this is the most crucial part of your disaster recovery plan, since it allows you to end the threat quickly and begin responding to it before it's too late.

Define roles and responsibilities.

Within your organization, make sure you define the roles and responsibilities of your staff members.

You already have one person in charge of overseeing the finalization and potential execution of your cybersecurity disaster recovery plan. But who will be responsible for coordinating resources on the ground level to execute that plan?

Additionally, who will be in charge of coordinating communication with stakeholders?

You don't want to be scrambling around at the last minute, wondering who's responsible for what. Secure organizations tend to run drills, so there's no ambiguity in internal roles and responsibilities. As a result, everyone knows what they're responsible for because they practiced it.

Invest in data backups.

Data backups are an indispensable tool in cybersecurity disaster recovery.

If all your data is securely backed up in an independent location, you'll have an option to restore your systems no matter what threats you're facing.

Ransomware attacks, DDoS attacks, and total corruption of your data won't cause permanent damage. You'll always be able to restore a previous version of your company's most important resources.

Create a response plan.

Of course, you'll also need to solidify the action items within your response plan. So, once you identify a threat, what will you do?

Prioritize business continuity.

Your biggest priority needs to be establishing business continuity.

What steps need to be taken to ensure that the business can continue serving customers without interruptions?

- Create alternative channels, services, and facilities.
- In pursuit of this, it's a good idea to document alternative channels, services, and facilities that your business can use.
- Assume your primary communication resources have been compromised.

How can you make a smooth transition?

- Put together a communication plan.

Think about how you're going to communicate with your internal team that the threat is underway.

- Plan how you're going to announce the threat to stakeholders and the general public.
- Track recovery metrics.
- Establish protocols for tracking recovery metrics.
- For example, how quickly did you respond to the threat once identified?
- Additionally, how much time did it take you to get the business back up and running?
- Document and reassess.
- Finally, you'll need to establish some protocols for documenting the threat.

Protocols include evaluating your disaster recovery execution and making improvements for the future. Good cybersecurity strategies always have an element of continuous improvement. There are always things that you can improve on and always new things to learn.

Don't assume that the cybersecurity disaster recovery plan you made three years ago is still relevant. But, hopefully, it's at least still reflective of your best work.

In conclusion, the more proactive you are with your company's cybersecurity strategy, the better protected you're going to be against a rising number of business threats in the digital space. Of course, with ample prevention, you may never have to use it. However, it will serve as an invaluable safety net in a worst-case scenario.

4.12 CRYPTOGRAPHY AND AUTHENTICAION

The word cryptography has come from a Greek word, which means secret writing. In the present day context it refers to the tools and techniques used to make messages secure for communication between the participants and make messages immune to attacks by hackers. For private communication through public network, cryptography plays a very crucial role. The role of cryptography can be illustrated with the help a simple model of cryptography as shown in Fig. 2.

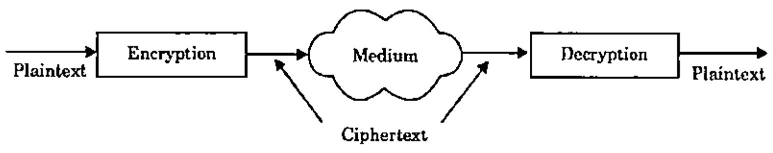


Fig. 2. A simple cryptography model.

Notes

The message to be sent through an unreliable medium is known as plaintext, which is encrypted before sending over the medium. The encrypted message is known as ciphertext, which is received at the other end of the medium and decrypted to get back the original plaintext message. In this lesson we shall discuss various cryptography algorithms, which can be divided into two broad categorize - Symmetric key cryptography and Public key cryptography. Cryptography algorithms based on symmetric key cryptography are presented in Sec. 6.2. Public key cryptography has been addressed in Sec. 6.3

4.13 SYMMETRIC KEY CRYPTOGRAPHY

The cipher, an algorithm that is used for converting the plaintext to ciphertex, operates on a key, which is essentially a specially generated number (value). To decrypt a secret message (ciphertext) to get back the original message (plaintext), a decrypt algorithm uses a decrypt key. In symmetric key cryptography, same key is shared, i.e. the same key is used in both encryption and decryption as shown in Fig. 3. The algorithm used to decrypt is just the inverse of the algorithm used for encryption. For example, if addition and division is used for encryption, multiplication and subtraction are to be used for decryption.

Symmetric key cryptography algorithms are simple requiring lesser execution time. As a consequence, these are commonly used for long messages. However, these algorithms suffer from the following limitations:

- Requirement of large number of unique keys. For example for n users the number of keys required is n(n-l)/2.
- Distribution of keys among the users in a secured manner is difficult.

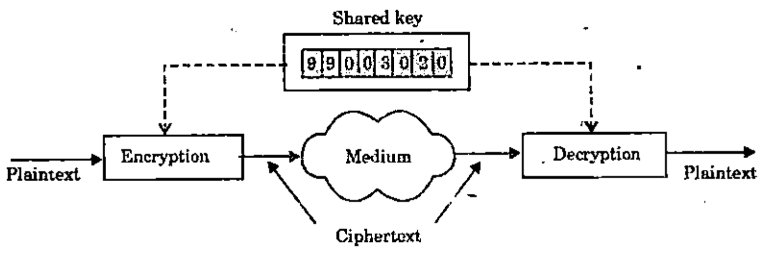


Fig. 3. A simple symmetric key cryptography model.

Monoalphabetic Substitution

One simple example of symmetric key cryptography is the Monoalphabetic substitution. In this case, the relationship between a character in the plaintext and a character in the ciphertext is always one-to-one.

An example Monoalphabetic substitution is the Caesar cipher. As shown in Fig. 4, in this approach a character in the ciphertext is substituted by another character shifted by three places, e.g. A is substituted by D. Key feature of this approach is that it is very simple but the code can be attacked very easily.

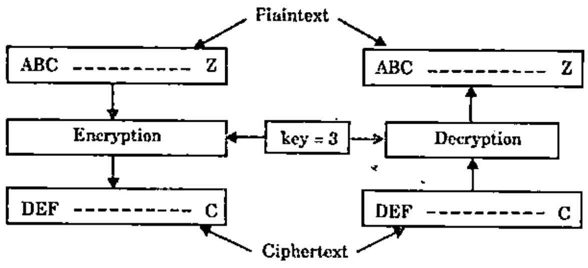


Fig. 4 The Caesar cipher.

Polyalphabetic Substitution

This is an improvement over the Caesar cipher. Here the relationship between a character in the plaintext and a character in the ciphertext is always one-to-many.

Example: Example of polyalphabetic substitution is the Vigenere cipher. In this case, a particular character is substituted by different characters in the ciphertext depending on its position in the plaintext. Fig. 5 explains the polyalphabetic substitution. Here the top row shows different characters in the plaintext and the characters in different bottom rows show the characters by which a particular character is to be replaced depending upon its position in different rows from row-0 to row-25.

- Key feature of this approach is that it is more complex and the code is harder to attack successfully.

Character in plaintext

	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0	W R K D O V C A S B Y Q M L H I T U F E Z N G J P X
1	H Q B G W E R K F C O A Z J M S L V N I P U D T X Y
2	P I D Z X V S T O C M J N L B Q R U W K H G E F A Y
⋮	⋮
25	M C I D A X V S T O N L K U R E W Z H F P G Y J B Q

Character in ciphertext

Fig. 5. Polyalphabetric substitution.

Transpositional Cipher

The transpositional cipher, the characters remain unchanged but their positions are changed to create the ciphertext. Fig. 6 illustrates how five lines of a text get modified using transpositional cipher. The characters are arranged in two-dimensional matrix and columns are interchanged according to a key is shown in the middle portion of the diagram.

Notes

The key defines which columns are to be swapped. As per the key shown in the Fig., character of column is to be swapped to column 3, character of column 2 is to be swapped to column 6, and so on. Decryption can be done by swapping in the reverse order using the same key.

Transpositional cipher is also not a very secure approach. The attacker can find the plaintext by trial and error utilizing the idea of the frequency of occurrence of characters.

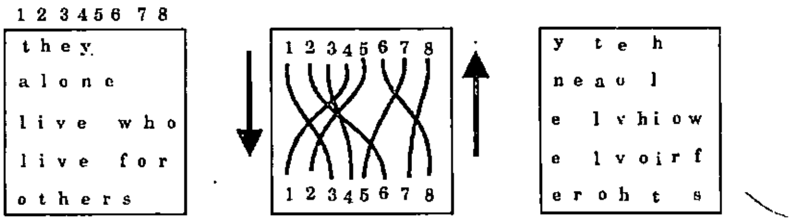


Fig. 6. Operation of a transpositional cipher.

Block Ciphers

Block ciphers use a block of bits as the unit of encryption and decryption. To encrypt a 64-bit block, one has to take each of the 264 input values and map it to one of the 264 output values. The mapping should be one-to-one. Encryption and decryption operations of a block cipher are shown in Fig. 7. Some operations, such as permutation and substitution, are performed on the block of bits based on a key (a secret number) to produce another block of bits. The permutation and substitution operations are shown in Fig. 6.7 and 6.8, respectively. In the decryption process, operations are performed in the reverse order based on the same key to get back the original block of bits

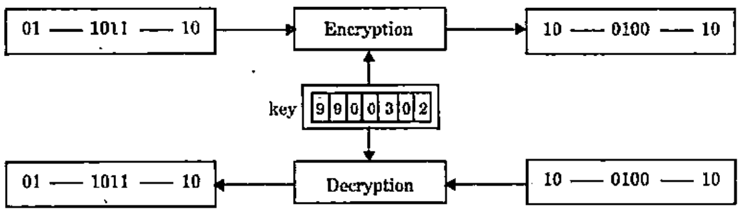


Fig. 7. Transformations in Block Ciphers.

Permutation: As shown in Fig. 8, the permutation is performed by a permutation box at the bit-level, which keeps the number of 0s and 1s same at the input and output. Although it can be implemented either by a hardware or a software, the hardware implementation is faster.

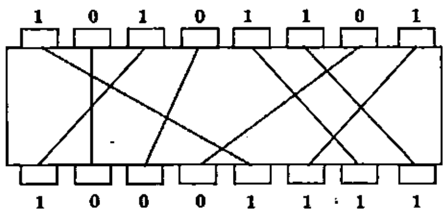


Fig. 8. Permutation operation used in Block Ciphers.

Substitution: As shown in Fig. 9, the substitution is implemented with the help of three building blocks - a decoder, one p-box and an encoder. For an n-bit input, the decoder produces an 2^n bit output having only one 1, which is applied to the P-box. The P-box permutes the output of the decoder and it is applied to the encoder. The encoder, in turn, produces an n-bit output. For example, if the input to the decoder is 011, the output of the decoder is 00001000. Let the permuted output is 01000000, the output of the encoder is 011.

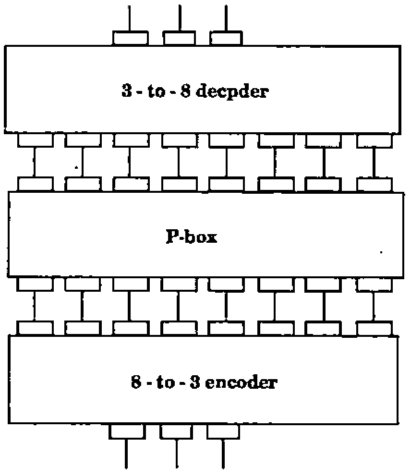


Fig. 10. Substitution operation used in Block Ciphers.

A block Cipher: A block cipher realized by using substitution and permutation operations is shown in Fig. 11. It performs the following steps:

Step-1: Divide input into 8-bit pieces

Step-2: Substitute each 8-bit based on functions derived from the key

Step-3: Permute the bits based on the key

All the above three steps are repeated for an optimal number of rounds.

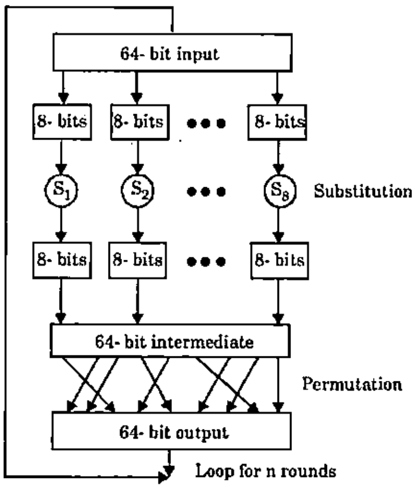


Fig. 11. Encryption by using substitution and permutation.

Notes

4.14 DATA ENCRYPTION STANDARD (DES)

One example of the block cipher is the Data Encryption Standard (DES). Basic features of the DES algorithm are given below:

- A monoalphabetic substitution cipher using a 64-bit character
- It has 19 distinct stages
- Although the input key for DES is 64 bits long, the actual key used by DES is only 56 bits in length.
- The decryption can be done with the same password; the stages must then be carried out in reverse order.
- DES has 16 rounds, meaning the main algorithm is repeated 16 times to produce the ciphertext.
- As the number of rounds increases, the security of the algorithm increases exponentially.
- Once the key scheduling and plaintext preparation have been completed, the actual encryption or decryption is performed with the help of the main DES algorithm as shown in Fig. 12.

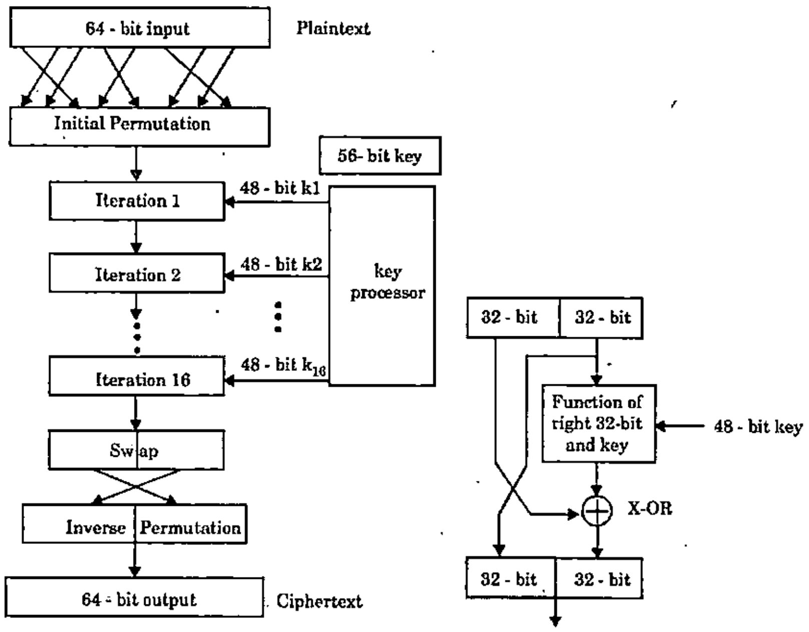


Fig. 12. 64-bit Data Encryption Standard (DES).

Encrypting a Large Message

DES can encrypt a block of 64 bits. However, to encrypt blocks of larger size, there exist several modes of operation as follows:

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)

- Cipher Feedback Mode (CFB)
- Output Feedback Mode (OFB)

Electronic Code Book (ECB)Electronic Code Book (ECB)

This is part of the regular DES algorithm. Data is divided into 64-bit blocks and each block is encrypted one at a time separately as shown in Fig. 13. Separate encryptions with different blocks are totally independent of each other.

Disadvantages of ECBDisadvantages of ECB

- If a message contains two identical blocks of 64-bits, the ciphertext corresponding to these blocks are identical. This may give some information to the eavesdropper
- Someone can modify or rearrange blocks to his own advantage
- Because of these flaws, ECB is rarely used

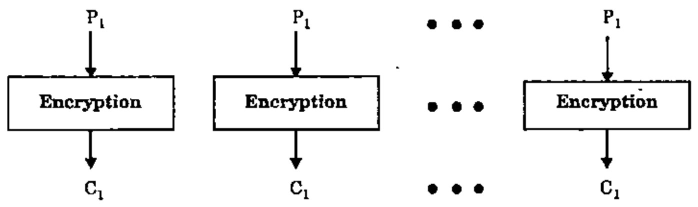


Fig. 13. Electronic Code Book (ECB) encryption technique.

Cipher Block Chaining (CBC)Cipher Block Chaining (CBC)

In this mode of operation, encrypted ciphertext of each block of ECB is XORed with the next plaintext block to be encrypted, thus making all the blocks dependent on all the previous blocks. The initialization vector is sent along with data as shown in Fig. 14.

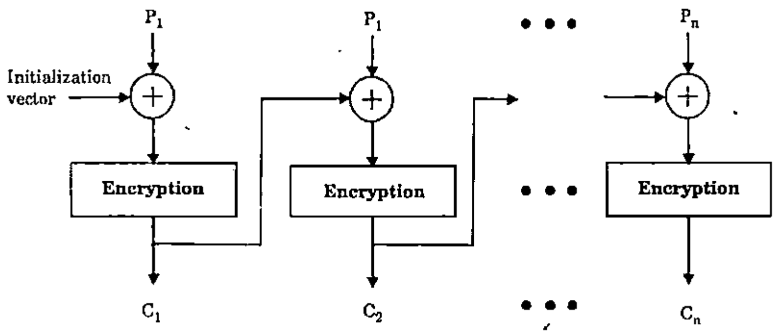


Fig. 14. Cipher Block Chaining (CBC) encryption technique.

Notes

Cipher Feedback Mode (CFB)

- In this mode, blocks of plaintext that is less than 64 bits long can be encrypted as shown in Fig. 15.
- This is commonly used with interactive terminals
- It can receive and send k bits (say k=8) at a time in a streamed manner 64 bit shift Register

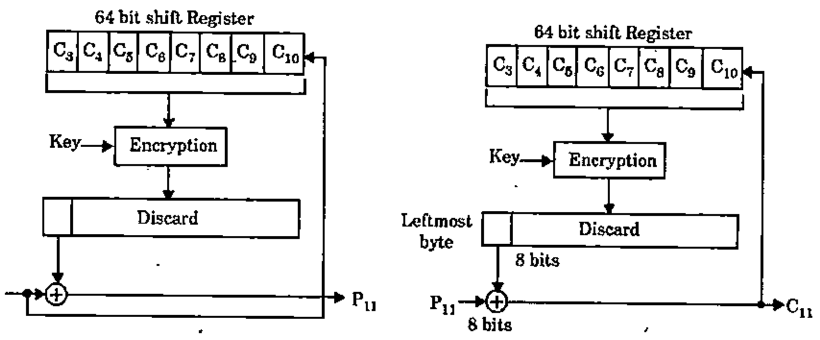


Fig. 15. Cipher Feedback Mode (CFB) encryption technique.

Output Feedback Mode (OFB)Output Feedback Mode (OFB)

The encryption technique of Output Feedback Mode (OFB) is shown in Fig. 16. Key features of this mode are mentioned below:

- OFB is also a stream cipher
- Encryption is performed by XORing the message with the one-time pad
- One-time pad can be generated in advance
- If some bits of the ciphertext get garbled, only those bits of plaintext get garbled
- The message can be of any arbitrary size
- Less secure than other modes

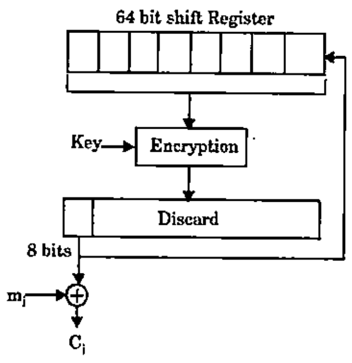


Fig. 16. Output Feedback Mode (OFB) encryption technique.

Triple DES

Triple DES, popularly known as 3DES, is used to make DES more secure by effectively increasing the key length. Its operation is explained below:

- Each block of plaintext is subjected to encryption by Kl, decryption by K2 and again encryption by K1 in a sequence as shown in Fig. 17
- CBC is used to turn the block encryption scheme into a stream encryption scheme

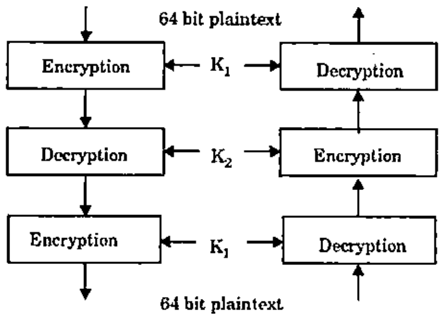


Fig. 17. Triple DES encryption technique.

Public key Cryptography

In public key cryptography, there are two keys: a private key and a public key. The public key is announced to the public, where as the private key is kept by the receiver. The sender uses the public key of the receiver for encryption and the receiver uses his private key for decryption as shown in Fig. 18.

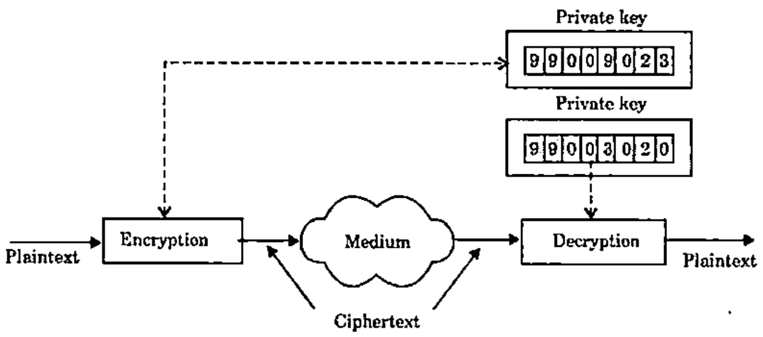


Fig. 18. Public key encryption technique.

Advantages:

- The pair of keys can be used with any other entity
- The number of keys required is small

Notes

Disadvantages

- . It is not efficient for long messages
- Association between an entity and its public key must be verified

RSA

The most popular public-key algorithm is the RSA (named after their inventors Rivest, Shamir and Adleman) as shown in Fig. 19. Key features of the RSA algorithm are given below:

- Public-key algorithm that performs encryption as well as decryption based on number theory
- Variable key length; long for enhanced security and short for efficiency (typical 512 bytes)
- Variable block size, smaller than the key length
- The private key is a pair of numbers (d, n) and the public key is also a pair of numbers (e, n)
- Choose two large primes p and q (typically around 256 bits) Compute n = p × q and z = (p – l) × (q – l) Choose a number d relatively prime to
- Find e such that e × d mod (p – l) × (q – l) =
- For encryption: C = Pe (mod n)
- For decryption: P = Cd (mod n)

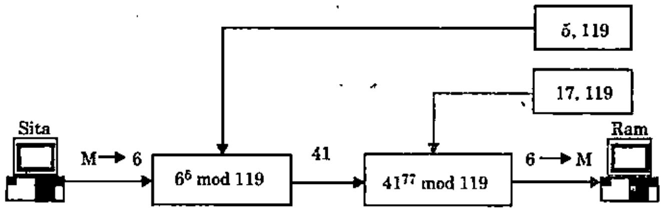


Fig. 19. The RSA public key encryption technique.

4.15 MANAGING RISK

"Tomorrow problems are today's risk." Hence, a clear definition of a "risk" is a problem that could cause some loss or threaten the progress of the project, but which has not happened yet.

These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale. Risk Management is the system of identifying addressing and eliminating these problems before they can damage the project.

We need to differentiate risks, as potential issues, from the current problems of the project. Different methods are required to address these two kinds of issues.

For example, staff storage, because we have not been able to select people with the right technical skills is a current problem, but the threat of our technical persons being hired away by the competition is a risk.

Risk is the likelihood that a loss will occur. Losses occur whena threat exposes a vulnerability. Organizations of all sizes facerisks. Some risks are so severe they cause a business to fail. Other risks are minor and can be accepted without anotherthought. Companies use risk management techniquesto identify and differentiate severe risks from minor risks.

When this is done properly, administrators and managers canintelligently decide what to do about any type of risk. The endresult is a decision to avoid, transfer, mitigate, or accept a risk.The common themes of these definitions are threat,vulnerability, and loss. Even though the common body ofknowledge (CBK) – see note – doesn't specifically mention loss,it implies it. Here's a short definition of each of these terms:

- Threat – A threat is any activity that representsa possible danger.
- Vulnerability – A vulnerability is a weakness.
- Loss – A loss results in a compromise to businessfunctions or assets.

Risks occur when threats exploit vulnerabilities, resulting in a loss. The loss cancompromise business functions and business assets. Losses also drive business costs.Risk management helps a company identify risks that need to be reduced. The firststeps in risk management are to identify threats and vulnerabilities. These can thenbe paired to help determine the severity of the risk.You can manage risks by choosing one of four techniques: A risk can be avoided,transferred, mitigated, or accepted. The primary risk management technique isrisk mitigation. Risk mitigation is also known as risk reduction or risk treatment.You reduce vulnerabilities by implementing control.

Risk management includes an evaluation of services you provide to customers. In thiscontext, a customer is any entity that receives a service. Obvious customers are those that purchase your services.For example, if your organization provides e-mail services tosmall businesses, these small businesses are your customers. Instead of managing their own e-mail servers, they outsourcethe service to you.

These customers have an expectation of the service. Theycould expect that e-mail is available 24 hours a day, seven daysa week. Alternatively, they may expect access to the e-mailonly during their business hours. Either way, it's importantto identify the expectations. A service level agreement (SLA) is a document that identifiesan expected level of performance. It identifies the minimumuptime or the maximum downtime.

Organizations use SLA sas a contract between a service provider and a customer. An SLA can identify monetary penalties if the terms aren't met.If your organization has SLAs with other organizations, these should be includedin the risk management review. You should pay special attention to monetary penalties.

For example, an SLA could specify a maximum downtime of four hours. After fourhours, hourly penalties will start to accrue. You can relate this to the MAO.Of course, SLAs that promise low levels of downtimes cost more. This extra cost is imposed to pay for the extra controls that are used. These extra controls providea higher level of service.A less obvious customer is the internal customer. Any employee or departmentthat receives a service is a customer

4.16 INFORMATION SECURITY POLICY

The basic objective is to communicate securely over an insecur e medium. Any action that compromises the security of information can be considered as attack on security. Possible type of attacks mentioned below:

- Interruption: It is an attack on the availability of information by cutting wires, jamming wireless signals or dropping of packets by a switch.
- Interception: As a message is communicated through a network, eavesdroppers can listen in use it for his/her own benefit and try to tamper it.
- Modification: As a message is communicated through a network, eavesdroppers can intercept it and send a modified message in place of the original one.
- Fabrication: A message may be sent by a stranger by posing as a friend. This is also known as impersonation.

These attacks can be prevented with the help of several services implemented with the help of cryptography, as mentioned in the following section.

4.17 SECURITY SERVICES

Secured communication requires the following four basic services:

- Privacy: A person (say Sita) should be able to send a message to another person (say Ram) p.rivately. It implies that to all others the message should be unintelligible.
- Authentication: After the message is received by Ram, he should be sure that the message has been sent by nobody else but by Sita.
- Integrity: Ram should be sure that the message has not been tampered on transit.
- Nonrepudiation: Ram should be able to prove at a later stage that the message was indeed received from Sita.

Privacy

Privacy can be achieved using symmetric key cryptography. In this case, the key is shared between the sender (Sita) and the receiver (Ram) as shown in Fig. 20. Privacy can also be achieved by using public-key cryptography as shown in Fig. 21. However, in this case the owner should be verified.

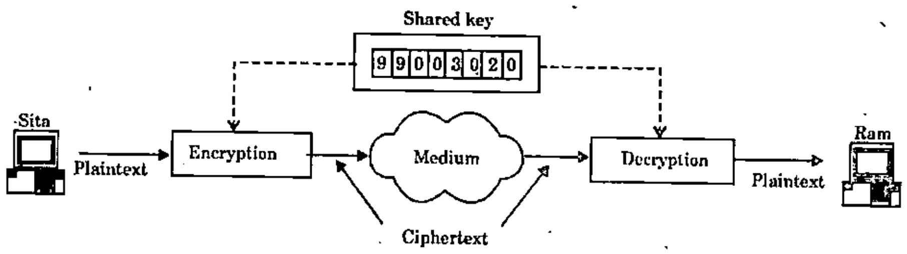


Fig. 20. Privacy using private-key cryptography.

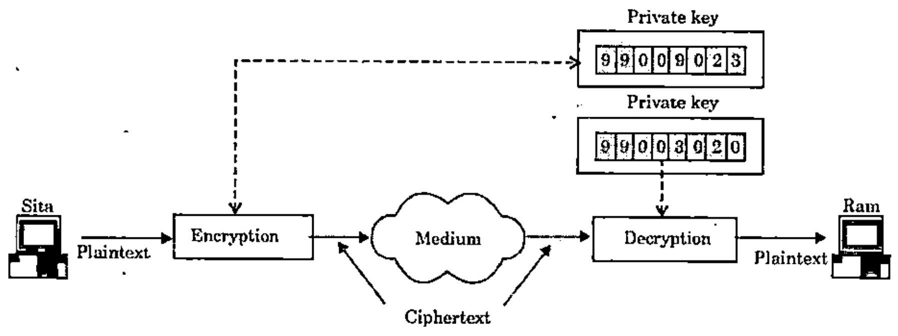


Fig. 21. Privacy using public-key cryptography.

4.18 AUTHENTICATION, INTEGRITY AND NONREPUDIATION USING DIGITAL SIGNATURE

By message authentication we mean that the receiver should be sure about sender's identity. One approach to provide authentication is with the help of digital signature. The idea is similar to signing a document. Digital Signature provides the remaining three security services; Authentication, Integrity and Nonrepudiation.

Digital Signature

There are two alternatives for Digital Signature:

- Signing the entire document
- Signing the digest

In the first case the entire document is encrypted using private key of the sender and at the receiving end it is decrypted using the public key of the sender as shown in Fig. 22. For a large message this approach is very inefficient. In the second case a miniature version of the message, known as digest, is encrypted using the private key of the sender and then the signed digest along with the message is sent to the receiver as shown in Fig. 23.

The receiver decrypts the signed digest using the public key of the sender and the digest created using the received message is compared with the decrypted digest as shown in Fig. 7.5. If the two are identical, it is assumed that the sender is authenticated. This is somewhat similar to error detection using parity bit.

Notes

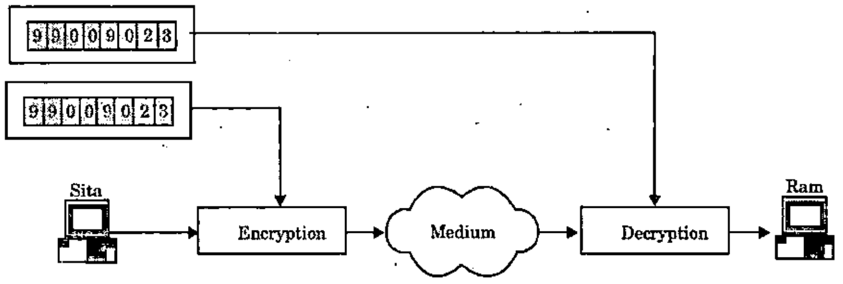


Fig. 22. Authentication by signing the whole document.

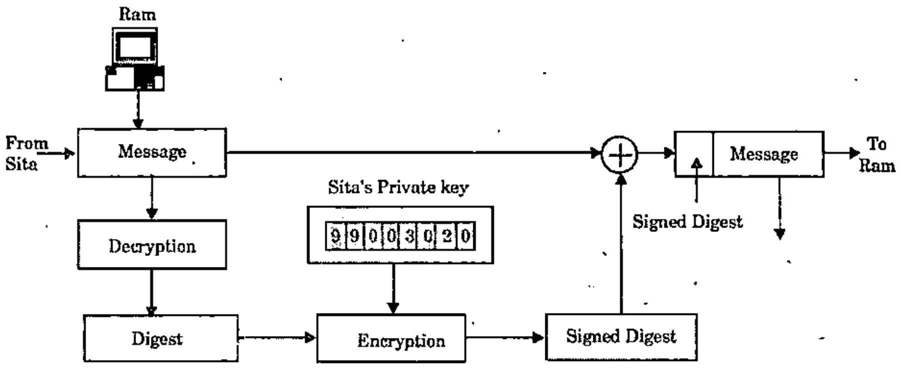
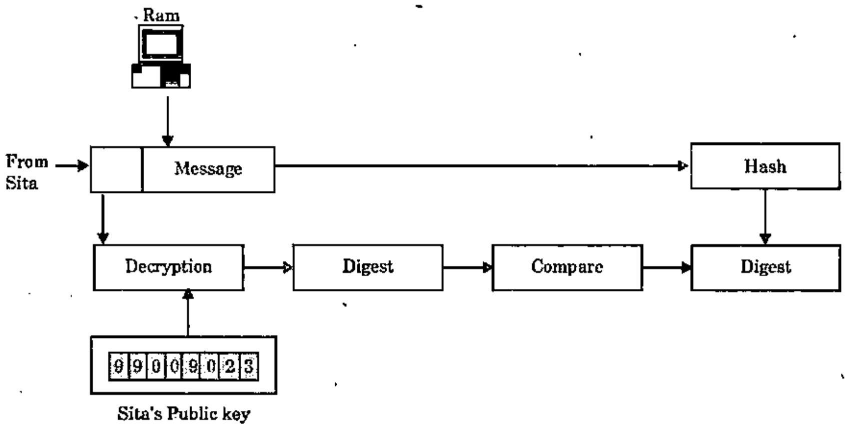


Fig. 23. Sender site for authentication by signed digest.



Some key features of this approach are mentioned below:

- Digital signature does not provide privacy
- Hash function is used to create a message digest

- It creates a fixed-length digest from a variable-length message

Most common Hash functions:

- m MD5 (Message Digest 5): 120-bit
- m SHA-1 (Secure Hash algorithm 1): 160-bit

Important properties:

- One-to-One
- One-way

4.19 USER AUTHENTICATION USING SYMMETRIC KEY CRYPTOGRAPHY

User authentication is different from message authentication. In case of message authentication, the identity of the sender is verified for each and every message. On the other hand, in user authentication, the user authentication is performed once for the duration of system access.

In the first approach, the sender (Sita) sends her identity and password in an encrypted message using the symmetric-key ksr and then sends the message as shown in Fig. 24. However, an intruder (say Ravana) can cause damage without accessing it. He can also intercept both the authentication message and the data message, store them and then resends them, which is known as replay attack.

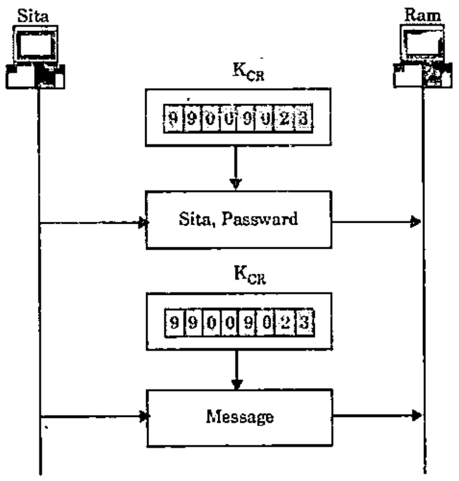


Fig. 24. User authentication using symmetric key cryptography.

- Using nonce, a large random number used only once
- To prevent the replay attack, the receiver (Ram) sends nonce, a large random number that is used only once to the sender (Sita) to challenge Sita. In response Sita sends an

Notes

encrypted version of the random number using the symmetric key. The procedure is shown in Fig. 25.

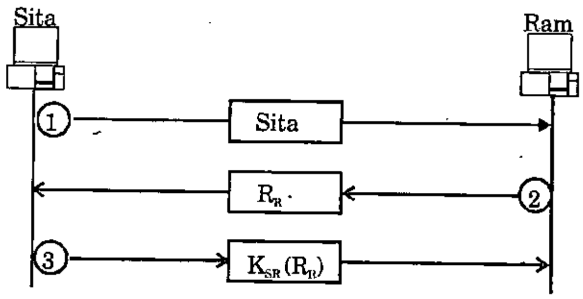


Fig. 25 User authentication using a nonce.

Bidirectional Authentication

In the bidirectional authentication approach, Ram sends nonce to challenge Sita and Sita in turn sends nonce to challenge Ram as shown in Fig. 26. This protocol uses extra messages for user authentication. Protocol with lesser number of messages is possible.

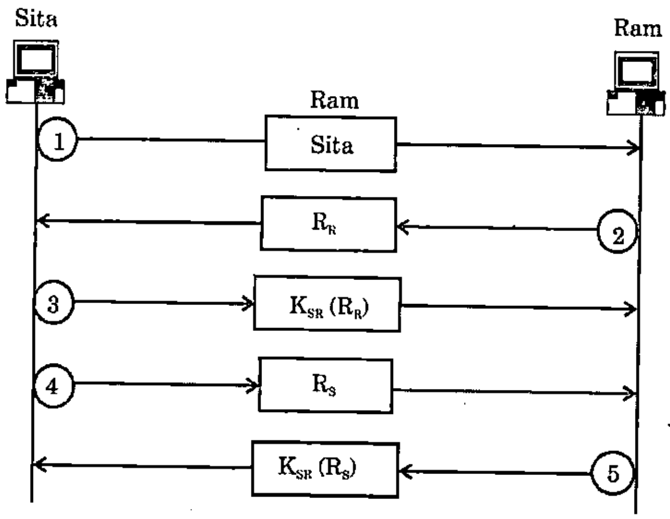


Fig. 26. Bidirectional authentication using a nonce.

4.20 USER AUTHENTICATION USING PUBLIC KEY CRYPTOGRAPHY

Public key cryptography can also be used to authenticate a user. The procedure is shown in Fig. 27.

ER = Public key of Ram,

Es = Public key of Sita

rs = nonce by Sita,

RR = nonce by Ram

Ks = Session key sent by Ram

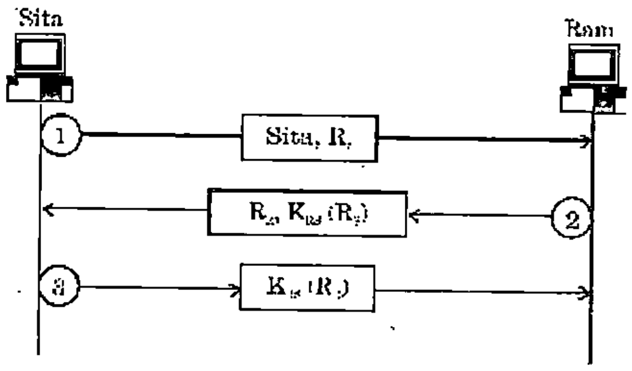


Fig. 27. User authenti cation using public key cryptography.

4.21 KEY MANAGEMENT

Although symmetric-key and public-key cryptography can be used for privacy and user authentication, question arises about the techniques used for the distribution of keys. Particularly, symmetric-key distribution involves the following three problems:

- For n people to communicate with each other requires n(n-l)/2 keys. The problem is aggravated as n becomes very large.
- Each person needs to remember (n-1) keys to communicate with the the remaining (n-1) persons.
- How the two parties will acquire the shared key in a secured manner?
- In view of the above problems, the concept of session key has emerged. A session key is created for each session and destroyed when the session is over. The Diffie-Hellman protocol is one of the most popular approach for providing one-time session key for both the parties.

Diffie-Hellman Protocol

Key features of the Diffie-Hellman protocol are mentioned below and the procedure is given in Fig. 28.

- Used to establish a shared secret key
- Prerequisite: N is a large prime number such that (N–1)/2 is also a prime number.
- G is also a prime number. Both N and G are known to Ram and Sita..
- Sita chooses a large random number x and calculates R1 = Gx mod N and sends it to Ram

- Ram chooses another large random number y and calculates R2 = Gymod N and sends it to Sita
- Ram calculates K = (R1)y mod N
- Sita calculates K = (R2)x mod N

4.22 APPLICATION LAYER SECURITY

Based on the encryption techniques we have discussed so far, security measures can be applied to different layers such as network, transport or application layers. However, implementation of security features in the application layer is far simpler and feasible compared to implementing at the other two lower layers. In this subsection, a protocol known as Pretty Good Privacy (POP), invented by Phil Zinmermann, that is used in the application layer to provide all the four aspects of security for sending an email is briefly discussed. POP uses a combination of private-key and public key for privacy. For integrity, authentication and nonrepudiation, it uses a combination of hashing to create digital signature and public-key encryption.

Virtual Private Network (VPN)

With the availability of huge infrastructure of public networks, the Virtual Private Network (VPN) technology is gaining popularity among enterprises having offices distributed throughout the country. Before we discuss about the VPN technology, let us first discuss about two related terms: intranet and extranet.

Intranet is a private network (typically a LAN) that uses the internet model for exchange of information. A private network has the following features:

- It has limited applicability because access is limited to the users inside the network
- Isolated network ensures privacy
- Can use private IP addresses within the private network

Extranet is same as the intranet with the exception that some resources can be allowed to access by some specific groups under the control of network administrator.

Privacy can be achieved by using one of the three models: Private networks, Hybrid Networks and Virtual Private Networks.

Private networks: A small organization with a single site can have a single LAN whereas an organization with several sites geographically distributed can have several LANs connected by leased lines and routers as shown in Fig. 7.14. In this scenario, people inside the organization can communicate with each other securely through a private internet, which is totally isolated from the global internet.

4.23 CREATING A SECURE ENVIRONMENT

Every day, as a part of my work at AlienVault, I talk to prospective clients. Many of them are trying to put together a security plan for their business. Most of the people I talk to are IT professionals who, like everyone else, are learning as they go.

During my time in IT and the security industry, I have seen almost every type of network you could imagine. Most of them made sense and could be explained and I could understand why they were built the way they were. Some, not so much. During the last 10 years especially, I have started compiling network drawings and information on the many ways that networks are designed and deployed.

The following list of bullet points are my recommendations to an IT manager or business leader if they consulted me on how to put together information technology for their business. Please remember this is a fairly generic list and there are tons of deviations to take into consideration when building a network and then protecting it.

1. Policies and Procedures

Policies and procedures are the cornerstones of your IT governance. This is the "what is going to happen and how is going to happen" of your security posture, and from the big picture your entire IT infrastructure. Creating a solid policy and procedure document or documents will provide your organization with an IT and security blueprint for your initial build, maintenance, management and remediation of issues. Solid policy and procedure manual(s) will also prepare the environment to work within any framework and meet compliance requirements.

2. Gateway Security

Gateway security is essential to keeping the bad guys out. There are a number of popular firewalls on the market that will provide excellent security at the gateway. The needs of the environment will dictate which firewall will work best.

For example, a high throughput environment with a large internal IP count might require a Next Generation Firewall (NGF) that runs only a few services on board and reserves the majority of resources for ingress-egress traffic. On the other hand, an environment that requires a very high level of security but has limited WAN bandwidth may be better suited for a UTM (Unified Threat Management) firewall which runs a number of services onboard. Traditionally it also utilizes significant resources for services like deep packet inspection (DPI), data loss prevention, (DLP), gateway antivirus, website filtering, email filtering and other high-end security services.

3. End Point Security

As the old saying goes... AntiVirus is DEAD!!! Not really.

Actually, antivirus is evolving and morphing like your favorite advanced persistent threat (APT) malware. A few years back the InfoSec industry started to break new ground on digging deeper into threats and breaches using threat intelligence in real-time to actively pursue malware based on heuristic data. Heuristic data became important as technology progressed to utilize behavioral analysis based on up-to-date threat intelligence.

These progressions in the industry gave rise to Endpoint Detection and Response (EDR), which is quickly morphing into a formidable companion to traditional antivirus and antiMalware protection. The very minimum that should be deployed into an environment includes a good reputable antivirus with antimalware capabilities, however, to get a definite head start on any compromises within the environment EDR is highly recommended.

4. Identity and Access Management (IAM) / Multi-Factor Authentication (MFA)

IAM and MFA are two entirely different technologies, and for good cause. As so many tech manufacturers are trying to get rid of the password, setting up good IAM services which work as initial authentication and then a reputable MFA service that is authenticated separately from the IAM service is essential.

IAM services range from Active Directory and LDAP (Lightweight Directory Access Protocol) Cloud LDAP, and authentication services like those provided with AWS IAM services, Microsoft Azure Active Directory services, and Google Directory services. There are tons of IAM services to choose from, and the environment will dictate the type of IAM services to utse.

The other part of the equation is MFA. The industry is full of MFA providers, from Google Authenticate to Yubikey and many others. Whether it's token based, hardware or biometric-based, MFA it is important to understand that the second form of authentication needs to be separate from the initial authentication system and it needs to be secure. For example, biometric authentication is very popular, but if it is simply used as a shortcut to enter an insecure password then it is not a secure solution.

Soft tokens that are received through Short Messaging Service (SMS), though very popular, do not provide the level of security that is often marketed. SMS in and of itself is insecure. Traditional SMS messages are sent in clear text and are subject to being intercepted or even having the SMS service broker hacked and the unencrypted messages being stolen and used to hack other larger targets. There have also been instances of "man in the device" attacks that have been used to steal tokens as they come in from the SMS broker.

IAM and MFA are probably the most important aspects of any threat posture, because not only does it control ingress authentication from the WAN, but it also validates and authenticates internal users requesting access to various resources.

5. Mobile Protection, Remote Access, and Virtual Private Networks (VPN)

Mobile devices are more popular than ever, especially as millennials become more prevalent in the workplace. This creates an especially unique situation for InfoSec pros who are tasked with securing modern environments.

The options for securing these environments is growing on an almost daily basis. From mobile device management (MDM) to wireless networks that prevent devices from connecting to the network unless they pass authentication and a scan to ensure the mobile device meets the preset requirements. Examples include not allowing 3rd party downloads outside of the prescribed manufacturers' store, ensuring anti-virus and anti-malware is installed and up to date and ensuring the mobile operating systems is updated to the prescribed revision range.

Something else to note more and more AV/AM and EDR manufacturers are making versions of their solutions to accommodate most mobile operating systems.

Remote access and VPN to the environment has always been a tricky topic. Anytime an employee wants to, or is required to connect to the environment for work purposes, most IT professionals pop a couple of Tylenol because they know a headache is coming.

Not all VPNs are created equally. Simple VPN connections can be made with a firewall or gateway router (even the cheap ones) with a simple handshake and a GRE tunnel to all the remote endpoints to pass traffic through the open VPN ports and into the environment. This is a minimal security solution and HIGHLY not recommended for a security-intensive environment.

The better solution would be to set up a better VPN concentrator or gateway firewall that can handle VPN tunnels. Set up IPSec connections. This is a bit more work because you have to load a security certificate on the gateway concentrator and on the IPSec software installed on the remote endpoint. However, the extra work is essential in creating a secure handshake and connection between the two devices. IPSec connections are great and they can be created with very fast speed and provide more security than a GRE connection.

However, I believe the IPSec connection running with AES256 and higher encryption over TLS are the most secure connections. It uses the most modern security type, an extremely high level of encryption, and requires a static RSA certificate to be installed on both endpoints. The same goes for static VPNs or site-to-site VPNs. The IPSec connection will create secure AES256bit or higher encryption for the SSL Tunnel as well as encrypting the payload with secure AES256 bit or higher encryption while in transit over the VPN connection.

6. Wireless Network Security

The wireless network industry has matured significantly over the last 10 to 15 years. In the old days, you set up an access point with a WEP security code and everything was great. Easy to deploy, easy to connect to and fast. Man those were the days. Then, in 2001, 3 researchers working at Berkeley produced a paper named "(In)Security of the WEP algorithm". They found the following flaws in WEP:

- Passive attacks to decrypt traffic based on statistical analysis.
- Active attack to inject new traffic from unauthorized mobile stations, based on known plaintext.
- Active attacks to decrypt traffic, based on tricking the access point.
- A dictionary-building attack that, after analysis of about a day's worth of traffic, allows real-time automated decryption of all traffic.

Furthermore, there are currently attack vectors on the TKIP. There are two attacks known against TKIP:

Beck-Tews attack

Ohigashi-Morii attack (which is an improvement on the Beck-Tews attack)

Both of the two attacks on the list only could decrypt small portions of data, compromising confidentiality. What they can't give you is access to the network. To give you an idea of how much data can be recovered, a single ARP frame would take around 14-17 minutes to get the plain text. Getting useful information with this type of attack is very improbable (but not impossible) considering the rate of recovery. The only attack known, besides flaws in the firmware of some routers, is brute-forcing the WPA key. Generally, the key is generated as follows:

Key = PBKDF2(HMAC–SHA1,passphrase, ssid, 4096, 256)

The algorithm takes the type of HMAC to be used, the passphrase, the ssid as salt, the number of iterations the password will be hashed and the final length of the generated hash. Considering this algorithm is meant to prevent hashed passwords from being broken, it can take a huge amount of time. The only reasonable attack would be to use a dictionary attack (hence it is important to use long passwords containing characters, numbers, and letters).

Also, note that you need to change your SSID to something very random. Rainbow tables have been generated for the top 1000 used SSIDs. Which can reduce attack time significantly.

WPA also supports AES (which can be used instead of RC4). While AES is more secure than RC4 the biggest problem of WPA is still present, namely, the integrity check is still done using TKIP-MIC.

Other things to consider when deploying a network system are things like:

- Ease of management
- Centralized or cloud-based management

How the rogue AP detection works. Does it go after non-system-based AP's as rogue or does the system utilize logic to decide which AP's are rogue and which AP's may be legally active depending on activity, SSID names and perhaps MAC addresses?

Please note there are FCC regulations against rogue AP detection and destruction.

Every manufacturer has their own unique functionality around rogue AP detection and many other security functions. Iit is advised that the purchaser and IT department perform due diligence before making a decision.

Other things to look at are authentication types, 802.1x, Active Directory, LDAP, AAA services are some of the more popular authentication types.

Lastly, it may be important to ensure that the system employs a guest WiFi authentication system. This will prevent, or at least deter, spoofing of the guest network as well as giving the WiFi owner an opportunity to collect information on every asset that connects to the guest WiFi. This is great for marketing and accountability purposes.

7. Back up and Disaster Recovery

Backup and disaster recovery (BDR) services are essential to an organization's incident planning to stay up and running in the event of a major catastrophe. BDR consists of backup and recovery of key IT systems and planning for the continuance of operations in the event that the organization encounters catastrophic events.

Such events may that cause the corporate or remote locations to become inoperable, destroyed or infected with destructive malware such as ransomware. There is an entire industry built around just back up and disaster recovery operations. It has been said about the growth of the industry as "this ain't your Daddy's old tape drive anymore".

When choosing a BDR service you absolutely need to ensure that they will meet the organization's needs 100%. Ensure that they provide a Service Level Agreement (SLA) of a minimum of 99.999% reliability. They should have a plan to return your data expeditiously. Lastly, the BDR firm must have a training and test plan to ensure the first time data is recovered for your environment is not during the catastrophe.

8. Environment Visibility

Security information and event management (SIEM) is an approach to security management that combines SIM (security information management) and SEM (security event management) functions into one security management system. The acronym SIEM is pronounced "sim" with a silent e.

A good SIEM system will provide the security team with detailed network and asset visibility, aggregation and parsing of all log files within the environment, and the ability to organize and search the log files in an organized way. The ability to do forensics in the event of an environment compromise is also required.

An exceptional system will provide all of the above-listed functions as well as threat Intelligence from a reputable threat intelligence community. The ability to correlate log files against threat intelligence to identify real-time threats within the environment is required.

There are many SIEM vendors on the market today, and each one has its own unique features and functions. No SIEM is a one-size-fits-all type of system. Some provide very granular functions but require a very high level of technical skills to deploy, setup, tune and maintain. Others may be much easier to set up, maintain and provide lots of features but may not be as granular.

In any case, great care should be taken when performing due diligence as the price point may vary - most SIEM systems are pretty expensive depending on a number of variables. SIEM systems are required for almost every major compliance requirement in the United States and many other countries. A SIEM could be the most powerful security tool in your arsenal and should make security analysis and remediation easier and faster, not difficult and more cumbersome.

9. Technical Training

Every website in the security industry has several articles about the shortage of good technical talent for the security industry.

Organizations now recognize that investment in security is a necessity. Yet with a current estimated 350,000 open cybersecurity positions in the US, and a predicted global shortfall of 3.5 million cybersecurity jobs by 2021 — according to Cybersecurity Ventures — the industry clearly has a massive problem regarding supply and demand.

As the old saying goes "it is better to hire attitude and train than to hire training and suffer the attitude". If you want your operation to run optimally, then you have,to train your people to make it do so. It is a smart move all the way around.

Notes

10. End User Security Awareness Training

Last but definitely not least. If you ask any security professional, IT person and most managers what the weakest link in any environment is - their answer will be a resounding "The End Users".

So how do you fix that? 90% of all end users want to do the right thing but they just don't know how. Most people do not come with the built-in skepticism to doubt everything and look for proof. Therefore you have hundreds of thousands of breaches each year, simply because an unknowing end user clicked on a link that downloaded a virus, malware or botnet. Train your end users and it will make your life easier.

While this is not a definitive or granular list of steps to take to deploy a very safe environment, it will definitely put you on the road to creating a secure environment that will enable the organization's IT and security staff to be proactive and shorten remediation times on almost any issue that they encounter.

4.24 INTERNET SECURITY STANDARDS

The need for security in all things technology is well-known and paramount. That includes the demand for the highest security standards in software development as well. For companies and developers, there is good news, as there are numerous security standards out there providing just those kind of guidelines and safeguards.

Security Standards

To make cybersecurity measures explicit, the written norms are required. These norms are known as cybersecurity standards: the generic sets of prescriptions for an ideal execution of certain measures. The standards may involve methods, guidelines, reference frameworks, etc. It ensures efficiency of security, facilitates integration and interoperability, enables meaningful comparison of measures, reduces complexity, and provide the structure for new developments.

A security standard is "a published specification that establishes a common language, and contains a technical specification or other precise criteria and is designed to be used consistently, as a rule, a guideline, or a definition." The goal of security standards is to improve the security of information technology (IT) systems, networks, and critical infrastructures. The Well-Written cybersecurity standards enable consistency among product developers and serve as a reliable standard for purchasing security products.

Security standards are generally provided for all organizations regardless of their size or the industry and sector in which they operate. This section includes information about each standard that is usually recognized as an essential component of any cybersecurity strategy.

ISO

ISO stands for International Organization for Standardization. International Standards make things to work. These standards provide a world-class specification for products, services and computers, to ensure quality, safety and efficiency. They are instrumental in facilitating international trade.

ISO standard is officially established On 23 February 1947. It is an independent, non-governmental international organization. Today, it has a membership of 162 national standards bodies and 784 technical committees and subcommittees to take care of standards development.

ISO has published over 22336 International Standards and its related documents which covers almost every industry, from information technology, to food safety, to agriculture and healthcare.

ISO 27000 Series

It is the family of information security standards which is developed by the International Organization for Standardization and the International Electrotechnical Commission to provide a globally recognized framework for best information security management. It helps the organization to keep their information assets secure such as employee'details, financial information, and intellectual property.

The need of ISO 27000 series arises because of the risk of cyber-attacks which the organization face. The cyber-attacks are growing day by day making hackers a constant threat to any industry that uses technology.

The ISO 27000 series can be categorized into many types. They are-

- **ISO 27001-** This standard allows us to prove the clients and stakeholders of any organization to managing the best security of their confidential data and information. This standard involves a process-based approach for establishing, implementing, operating, monitoring, maintaining, and improving our ISMS.
- **ISO 27000-** This standard provides an explanation of terminologies used in ISO 27001.
- ISO 27002- This standard provides guidelines for organizational information security standards and information security management practices. It includes the selection, implementation, operating and management of controls taking into consideration the organization's information security risk environment(s).
- **ISO 27005-** This standard supports the general concepts specified in 27001. It is designed to provide the guidelines for implementation of information security based on a risk management approach. To completely understand the ISO/IEC 27005, the knowledge of the concepts, models, processes, and terminologies described in ISO/IEC 27001 and ISO/IEC 27002 is required. This standard is capable for all kind of organizations such as non-government organization, government agencies, and commercial enterprises.
- **ISO 27032-** It is the international Standard which focuses explicitly on cybersecurity. This Standard includes guidelines for protecting the information beyond the borders of an organization such as in collaborations, partnerships or other information sharing arrangements with clients and suppliers.

IT Act

The Information Technology Act also known as ITA-2000, or the IT Act main aims is to provide the legal infrastructure in India which deal with cybercrime and e-commerce. The IT Act is based on the United Nations Model Law on E-Commerce 1996 recommended by

Notes

the General Assembly of United Nations. This act is also used to check misuse of cyber network and computer in India. It was officially passed in 2000 and amended in 2008. It has been designed to give the boost to Electronic commerce, e-transactions and related activities associated with commerce and trade. It also facilitate electronic governance by means of reliable electronic records.

IT Act 2000 has 13 chapters, 94 sections and 4 schedules. The first 14 sections concerning digital signatures and other sections deal with the certifying authorities who are licenced to issue digital signature certificates, sections 43 to 47 provides penalties and compensation, section 48 to 64 deal with appeal to high court, sections 65 to 79 deal with offences, and the remaining section 80 to 94 deal with miscellaneous of the act.

Copyright Act

The Copyright Act 1957 amended by the Copyright Amendment Act 2012 governs the subject of copyright law in India. This Act is applicable from 21 January 1958. Copyright is a legal term which describes the ownership of control of the rights to the authors of "original works of authorship" that are fixed in a tangible form of expression.

An original work of authorship is a distribution of certain works of creative expression including books, video, movies, music, and computer programs. The copyright law has been enacted to balance the use and reuse of creative works against the desire of the creators of art, literature, music and monetize their work by controlling who can make and sell copies of the work.

The copyright act covers the following-

- Rights of copyright owners
- Works eligible for protection
- Duration of copyright
- Who can claim copyright

The copyright act does not covers the following-

Ideas, procedures, methods, processes, concepts, systems, principles, or discoveries

- Works that are not fixed in a tangible form (such as a choreographic work that has not been notated or recorded or an improvisational speech that has not been written down)
- Familiar symbols or designs
- Titles, names, short phrases, and slogans
- Mere variations of typographic ornamentation, lettering, or coloring

Patent Law

Patent law is a law that deals with new inventions. Traditional patent law protect tangible scientific inventions, such as circuit boards, heating coils, car engines, or zippers. As time increases patent law have been used to protect a broader variety of inventions such as business practices, coding algorithms, or genetically modified organisms. It is the right to exclude others from making, using, selling, importing, inducing others to infringe, and offering a product specially adapted for practice of the patent.

In general, a patent is a right that can be granted if an invention is:

- Not a natural object or process
- New
- Useful
- Not obvious.

IPR

Intellectual property rights is a right that allow creators, or owners of patents, trademarks or copyrighted works to benefit from their own plans, ideas, or other intangible assets or investment in a creation.

These IPR rights are outlined in the Article 27 of the Universal Declaration of Human Rights. It provides for the right to benefit from the protection of moral and material interests resulting from authorship of scientific, literary or artistic productions. These property rights allow the holder to exercise a monopoly on the use of the item for a specified period.

Cyber Security Standards Overview

Cyber security standards are proliferating. Governments and businesses increasingly mandate their implementation.

More manufacturers and vendors are building and selling standards-compliant products and services. In addition, a growing number of organizations are becoming involved in standards development. Cyber security standards are being embraced because they are useful. They provide tangible benefits that justify the time and financial resources required to produce and apply them.

Security technology has not kept pace with the rapid development of IT, leaving systems, data, and users vulnerable to both conventional and innovative security threats. Politically motivated adversaries, financially motivated criminals, mischievous attackers, and malicious or careless authorized users are among the threats to systems and technology that have the potential to jeopardize cyber security, US economic security, consumer identities and privacy, and US public health and safety. While it is impossible to eliminate all threats, improvements in cyber security can help manage security risks by making it harder for attacks to succeed and by reducing the effect of attacks that do occur.

Cyber security standards enhance security and contribute to risk management in several important ways. Standards help establish common security requirements and the capabilities needed for secure solutions. For example, Federal Information Processing Standards (FIPS) 140-2, Security Requirements for Cryptographic Modules, establishes standard requirements for all cryptographic-based security systems used by federal organizations to protect sensitive or valuable data. Conformance testing can then be performed against the standard to provide assurance to users, that cryptographic modules are built to requirements.

Notes

Security standards facilitate sharing of knowledge and best practices by helping to ensure common understanding of concepts, terms, and definitions, which prevents errors. For example, the Information and Communications Technology (ICT) Security Standards Roadmap includes references to several security glossaries, including the ISO/IEC JTC1/SC27 IT Security Terminology publication. Other helpful resources include the Internet Security Glossary from the Internet Engineering Task Force (IETF), Request for Comments (RFC) 4949 and a compendium of the International Telecommunications Union Telecommunication Standardization Sector (ITU-T) approved security definitions. Using common definitions for security terminology saves time in the development of new standards and supports the interoperability of standards.

Cyber security standards also provide other benefits. Because standards generally incorporate best practices and conformance requirements, their use typically results in improvements in quality. Standards reduce the number of technical variations and allow consumers easy access to interchangeable technology.

Standards compliance programs offer a way to measure products and services against objective criteria and provide a basis for comparing products, such as confirming that they offer certain sets of security features. Consumers often benefit from cost savings that result from the development, manufacture, sales, and delivery of standards-based, interoperable products and services. Another benefit of cyber security standards is that the standards development process, with its typical practices of involving a wide range of subject-matter experts, prototyping, and incorporating conformity assessment criteria and methodologies, helps ensure that standards are implementable and reflect recommended practices. Products or services that have been demonstrated to conform to IT security standards can then be expected to offer more assurance than nonstandard products.

When security standards are not available for a technology, several problems often occur. Organizations that adopt the technology may not be aware of its inherent security weaknesses and the implications of implementing the technology for the organization's security posture. Organizations also may not have reliable information on how to take advantage of the technology's security capabilities or on what additional security controls may be needed to compensate for weaknesses in those capabilities. This tends to lead to insecure implementations and insufficient security maintenance, making systems more likely to be exploited and the organization more vulnerable to harm.

Cyber Security Standards Characteristics

Standards can be defined as widely used rules or specifications for activities or their results. Nevertheless, there are often significant differences in how individual standards are developed and applied. These differences can help determine how quickly and easily a new standard is embraced and thereby influence the continued use or demise of alternatives. As a result, standards are often described by the specific characteristics of their development and intended application, including the development process used to produce the standard, the way in which the standard is regulated, the applicability of the standard to different audiences, the availability of the standard to the public, and the measurability of the standard.

Standards come into being in different ways. Proprietary or company standards are developed by companies with little or no participation by external parties. De facto standards are created through the informal adoption of prevailing practices or norms. The majority are voluntary standards developed through some form of voluntary consensus process, in which stakeholders participate and agree. Some voluntary standards development efforts are open to all interested parties, while others are restricted to specific groups or individuals, such as members of a particular alliance or consortium.

Notes

Standards differ in the ways that they are regulated. Compliance with standards may be optional, or a governing or regulatory organization may make compliance a requirement. Voluntary standards are generally called voluntary, not only because they are created through volunteers' efforts but also because they are intended for optional use, although a regulating agency could adopt or mandate their use.

Mandatory standards are standards whose use is prescribed by a regulatory agency or implementing organization. Mandatory standards typically implement laws and regulations.

The audience to whom a standard applies depends upon the entity that develops or adopts it. An international standard is one that is adopted by an international standards development organization (SDO) and made available to the public, such as ISO International Standards. A regional standard is a standard adopted by several nations in a particular geographic region, for example, European Committee for Standardization (CEN) standards. A national standard is a standard developed for use in a particular country either by a government entity or a national SDO. A national standard can also be an international standard that is adopted for use by an individual country.

Examples include FIPS and American National Standards Institute (ANSI) standards in the United States and British Standards Institution (BSI) Standards in the United Kingdom. An industry standard is one that has been adopted by a particular industry for common use, for example, Security Industry Association (SIA) standards. Finally, a company standard, also known as a proprietary standard, is a standard developed and owned by a commercial entity that specifies practices or conventions unique to that entity.

Standards may or may not be freely accessible by everyone. By definition, open standards are publicly available, but their developer may charge for copies. Examples of open standards that are available to the public for a fee are ISO standards and standards developed by ANSI-accredited organizations. A vendor who develops and owns a proprietary standard may choose to make it available to promote interoperability and broaden the market, or choose not to share it.

A growing number of standards require a demonstration of conformance. A performance standard states requirements in terms of required results with criteria for verifying conformance, but without stating the methods for achieving required results.

Examples of performance standards are the Federal Information Security Management Act (FISMA) and the Health Insurance Portability and Accountability Act (HIPAA). A prescriptive standard specifies design requirements, how a requirement is to be achieved, or how an item is to be constructed, but without criteria for measuring the conformity of the results with specified requirements.

An example of a prescriptive standard is ISO/IEC 7810 on identification card physical construction. The development and use of performance standards are encouraged since they are less likely than prescriptive specifications to stand in the way of innovation. Still,

Notes

prescriptive standards are sometimes more appropriate, particularly for describing test methods or procedures or for defining standards to achieve interoperability.

4.25 CYBER SECURITY STANDARDS INTERACTION

A standard is rarely applied in isolation. When technologies, processes, and management practices are combined to solve a business problem, multiple standards normally come into play. When components are integrated, each may entail one or more technical or management standards. For example, a given business solution is likely to involve a variety of IT security configuration standards, such as networking, communications, and security-management standards. Each standard imposes requirements that may or may not conflict with the requirements of other standards.

Standards can interact in several ways. Some standards are complementary, which means that one standard supports or reinforces the requirements of another. For example, ISO frequently publishes multipart standards that can be considered complementary, where each part is a separately developed volume covering a different aspect of a centralissue. Some standards may conflict with each other, which means that there are inconsistencies or contradictions between standards, resulting in issues such as technological incompatibility or legal noncompliance.

Other standards are discrete, which means that they have no direct effect on one another. There are also standards gaps, where there is no formal standard developed for a particular area of security, although a guideline may exist. Standards gaps typically occur when a technology is evolving so rapidly that standards development cannot keep pace.

In other cases, a gap exists because consensus has not been reached on either the technology or the standard.

Standards and Guidelines

Standards can be contrasted with another category of documents, generally referred to as guidelines. Both standards and guidelines provide guidance aimed at enhancing cyber security, but guidelines usually lack the level of consensus and formality associated with standards. Some standards, such as ANSI Standards and FIPS Publications, are easily recognized because they include the term standard in their titles. Others are harder to recognize. For example, standards issued by the International Telecommunications Union (ITU), an international standards developer, are designated as Recommendations. A standard issued by the IETF starts out as an RFC and retains that designation even after being adopted as a standard. In other cases, documents that are not standards in the strict sense of the word may be treated as such by an organization if it suits the organization's needs.

For example, many US and international organizations and businesses have adopted National Institute of Standards and Technology (NIST) Special Publications as standards, even though those documents are published as guidelines for use by US Federal agencies.

Some organizations develop both standards and guidelines. For example, in addition to international standards, ISO/IEC issues several types of guidelines, including technical specifications, publicly available specifications (PAS), and technical reports, according to the

ISO/IEC Directives, Part 1, Section 3. A technical specification may be published when the immediate release of an international standard is not feasible, such as when the subject in question is still under development. A PAS may be an intermediate specification published prior to the development of a full international standard, or in International Electrotechnical Commission (IEC) it may be a "dual logo" publication published in collaboration with an external organization. A PAS does not fulfill the requirements for a standard. A technical report is an informative document generally intended to educate the reader, not to specify an international standard.

Cyber Security Standards Developers

International, regional, national, industry, and government groups are involved in the development of cyber security standards. An SDO is an organization whose primary mission is the development of voluntary consensus standards on an international, regional, or on a national basis. Most SDOs cover a wide variety of technical areas, not just cyber security. Consortia, industry alliances, and associations are all groups of organizations or individuals with similar interests that promote standards development. A consortium is typically formed for a limited time to achieve a specific goal, such as the development of standards. Industry alliances and associations tend to be more loosely formed to foster common interests. Consortia and industry alliances comprise companies, and associations are made up of individuals.

Finally, the US government and other national governments develop standards specifically intended for government audiences. Examples of organizations in each of these categories are provided below, along with brief discussions of some of the organizations' cyber security standards work.

International Standards Development Organizations

The Institute of Electrical and Electronics Engineers Standards Association (IEEE-SA) develops standards in many areas, including information technology, telecommunications, and power generation. An example of IEEE-SA's security work is its 802 Local Area Network (LAN)/ Metropolitan Area Network (MAN) Standards Committee.

Various working groups within the committee develop widely used standards for many types of networking technologies, such as Ethernet, wireless LANs, Bluetooth, and Worldwide Interoperability for Microwave Access (WiMAX). These standards include the security features built into the wireless networking protocols.

The IETF is concerned with the evolution of the Internet architecture and the operation of the Internet. The IETF has dozens of working groups that each focus on a different element of the Internet, including several groups working on Internet security. Topics addressed by these working groups include Domain Name System (DNS) security, authentication protocols, routing protocol security, Internet Protocol (IP) version 6, public key infrastructure, e-mail security, event logging, and network traffic encryption.

ISO, whose membership consists of the national standards institutes of more than 150 countries, addresses all standards except those for electrical and electronic engineering, which are the responsibility of the IEC. ISO and IEC formed the Joint Technical Committee 1 (JTC1) for IT standards development, including standards for the security of systems and

Notes

information. JTC1 has a number of subcommittees (SC) and working groups that address specific technologies. For example, the SC17 group addresses identification cards and personal identification, the SC27 group focuses on IT security techniques, the SC31 group works on automatic identification and data capture (AIDC) techniques, and the SC37 group develops biometric standards.

The ITU-T produces standards, called Recommendations, for telecommunication networks. ITU-T's standards are developed by study groups (SG) such as SG17, which covers security, languages, and telecommunications software.

SG17 led the development of the ICT Security Standards Roadmap, which provides information on previous and current security standards work from several major standards developers, including ISO, IEC, IETF, Organization for the Advancement of Structured Information Standards (OASIS), Institute of Electrical and Electronics Engineers (IEEE), and telecommunications-specific organizations. It also lists current security standards gaps and provides pointers to security glossaries. SG17 developed the ICT Security Standards Roadmap in collaboration with the European Network and Information Security Agency (ENISA) and the Network and Information Security Steering Group (NISSG).

Regional Standards Development Organizations

The European Telecommunications Standards Institute (ETSI) produces telecommunications standards within Europe. ETSI's cyber security standards activities include work on electronic signatures, smart cards, lawful interception, and 3GPP.

The CEN, whose members are the national standards organizations of 30 European countries, develops cyber security standards on its own and in conjunction with other international, national, and government standards developers.

National Standards Development Organizations

In the narrowest sense of the term, ANSI is not an SDO, since it does not develop standards; rather, it administers and coordinates the activities of the US private sector voluntary standardization system. ANSI sponsors cyber security-related working groups, such as a Homeland Security Standards Panel and a Healthcare Information Technology Standards Panel.

The InterNational Committee for Information Technology Standards (INCITS) is an ANSI-accredited organization, which develops US standards for information and communications technologies. INCITS comprise technical committees (TCs) that create standards for different technology areas. Examples of cyber security-focused TCs are B10 (identification cards and related devices), CS1 (cyber security), M1 (biometrics), and T6 (radio frequency identification (RFID) technology).

Consortia, Industry Alliances, and Associations

The Association for Automatic Identification and Mobility (AIM) is a trade association for entities that are interested in AIDC technologies. AIM performs the development of cyber security standards in areas such as barcodes, card technologies, electronic article surveillance, RFID, real-time locating systems (RTLS), and other AIDC-related technologies.

The British Security Industry Association (BSIA) is the professional trade association for the security industry in the United Kingdom. The BSIA develops codes of practice and technical documents and submits some of them for consideration as British Standards. Security areas addressed by the BSIA include access control; information destruction, physical security equipment, and security systems.

The Information Systems Audit and Control Association (ISACA) is an organization for information assurance, governance, security, and audit professionals. It is best known for its information system auditing and control standards and related initiatives.

For example, ISACA has developed Control Objectives for Information and relatedm Technology (COBIT), which is a control framework that encompasses several aspects of IT governance, including risk assessment. COBIT is based on various international standards and can be used to identify appropriate standards references during audits.

The Instrumentation, Systems, and Automation Society (ISA) is a professional association that develops standards for automation technologies. For example, its SP99 working group develops security standards for manufacturing and control systems, such as supervisory control and data acquisition (SCADA) systems and distributed control systems (DCS). Some of ISA's reports on this topic have become ANSI standards.

The OASIS develops standards for security and e-business, and is well known for its web services standards work. OASIS has several working groups focused on security topics such as biometrics, digital signature services, enterprise key management infrastructures, public key infrastructure adoption, and web services security.

The SIA is an ANSI-accredited SDO that develops systems integration and equipment performance standards. Several SIA working groups develop physical security standards on topics such as biometrics, mobile security devices, credential readers, security communications, and security control panels.

US Government Standards Developers

NIST develops security standards for US Federal information systems. NIST's FIPS have been made mandatory for federal use. Examples of FIPS include FIPS 200, which specifies minimum security requirements for federal information systems; FIPS 199, which provides standards for security categorization of federal systems; and FIPS 197, which defines the Advanced Encryption Standard (AES). NIST also hosts the National Center for Standards and Certification Information (NCSCI), which provides information on US standards and technical regulations, as well as other national, regional, and international standards.

The Office of Management and Budget (OMB) assists the President of the United States in the development of budget, management, and regulatory policies. OMB's products include OMB Circulars and OMB Memoranda, which are instructions or information issued to federal agencies.

Some of these documents mandate the use of particular security standards or require federal agencies to meet other security requirements. For example, OMB Circular A-130 pertains to the management of federal information resources, and OMB Memorandum M-07-16 mandates security controls to protect the confidentiality of personally identifiable information.

Notes

Getting Involved in Standards Development

In addition to those mentioned above, there are many other cyber security standards developers already working on creating new standards. The ICT Security Standards Roadmap provides information on a number of ongoing standards activities. Organizations interested in cyber security standards development can join existing standards efforts so that they can ensure that standards are developed in a way that is favorable to, or at least compatible with, their critical interests.

In addition to influencing the direction that a standard takes, actively participating in the standards development process offers other advantages. An organization gains a better understanding of the standards under development, their underlying designs, the trade-offs and compromises made during their development, and the operating conditions and environments they are intended to serve. Organizations make contacts and build relationships with technical experts involved in the development effort, as well as improving their own technical knowledge.

Participation in standards development also benefits the security community by sharing the effort across many organizations.

Most organizations do not participate in standards development activities. They may feel that it is not important, that it is impossible to influence the outcome, or that involvement is too expensive. Nevertheless, participation can be critical to realizing the benefits of standards. Also, organizations that choose not to get involved can find themselves faced with new standards with which they are not prepared to comply.

There are a number of ways to participate in the standards development process, each with its own level of resource commitment. Organizations can choose how fully to participate, depending upon the importance of the standard to the organization and the resources they have available to commit to the effort. Trackers follow the development of a standard at a high level, for example, by reading summaries and implementation timelines on the developer's public website.

Tracking the progress of a new standard gives organizations the ability to anticipate its effects, even if they choose not to become more actively involved in its development. Public reviewers review drafts of the standard and submit comments, which can influence the content and impact of a standard under development. For particularly important standards, organizations should consider becoming members of the entity developing the standards.

The role of driver may be appropriate when the organization's stake in a new standard is critical. It may be that producing the standard is part of the organization's charter or mission; driving the development of a standard may require significant resources.

In addition to contributing to the development of new standards, organizations should also consider participating in the maintenance of existing standards. Most standards undergo periodic review and revision. SDOs typically have established formal maintenance programs to help ensure that their standards do not become dated due to technological evolution, changes to related standards, or other causes.

SUMMARY

- Three basic security concepts important to information on the internet are confidentiality, integrity, and availability. Concepts relating to the people who use that information are authentication, authorization, and nonrepudiation.
- Information can be corrupted when it is available on an insecure network. When information is modified in unexpected ways, the result is known as loss of integrity. This means that unauthorized changes are made to information, whether by human error or intentional tampering.
- Information security is vital in an era in which data regarding countless individuals and organizations is stored in a variety of computer systems, often not under direct control. It is important to remember that security and productivity are often diametrically opposing concepts, and that being able to point out exactly when people are secure is a difficult task.
- Computer virus has the tendency to make its duplicate copies at a swift pace, and also spread it across every folder and damage the data of your computer system.
- A computer virus is actually a malicious software program or "malware" that, when infecting your system, replicates itself by modifying other computer programs and inserting its own code.
- Risk is the likelihood that a loss will occur. Losses occur when a threat exposes a vulnerability.
- A software project can be concerned with a large variety of risks. In order to be adept to systematically identify the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.
- Organizations of all sizes facerisks. Some risks are so severe they cause a business to fail. Other risks are minor and can be accepted without another thought. Companies use risk management techniques to identify and differentiate severe risks from minor risks. When this is done properly, administrators and managers can intelligently decide what to do about any type of risk.
- The end result is a decision to avoid, transfer, mitigate, or accept a risk. The common themes of these definitions are threat, vulnerability, and loss. Even though the common body of knowledge (CBK)— see note —doesn't specifically mention loss, it implies it.
- Risks occur when threats exploit vulnerabilities, resulting in a loss. The loss cancompromise business functions and business assets. Losses also drive business costs.Risk management helps a company identify risks that need to be reduced. The first steps in risk management are to identify threats and vulnerabilities.
- These can thenbe paired to help determine the severity of the risk. You can manage risks by choosing one of four techniques: A risk can be avoided,transferred, mitigated, or accepted. The primary risk management technique isrisk mitigation. Risk mitiga-

Notes

tion is also known as risk reduction or risk treatment.You reduce vulnerabilities by implementing control.

- Risk management includes an evaluation of services you provide to customers. In this context, a customer is any entity that receives a service. Obvious customers are those that purchase your services. For example, if your organization provides e-mail services tosmall businesses, these small businesses are your customers. Instead of managing their own e-mail servers, they outsourcethe service to you. These customers have an expectation of the service. They could expect that e-mail is available 24 hours a day, seven daysa week.
- Alternatively, they may expect access to the e-mailonly during their business hours. Either way, it's importantto identify the expectations.
- A service level agreement (SLA) is a document that identifiesan expected level of performance. It identifies the minimumuptime or the maximum downtime. Organizations use SLA sas a contract between a service provider and a customer. An SLA can identify monetary penalties if the terms aren't met. If your organization has SLAs with other organizations, these should be includedin the risk management review. You should pay special attention to monetary penalties. For example, an SLA could specify a maximum downtime of four hours.
- After four hours, hourly penalties will start to accrue. You can relate this to the MAO. Of course, SLAs that promise low levels of downtimes cost more. This extra costis imposed to pay for the extra controls that are used. These extra controls providea higher level of service.A less obvious customer is the internal customer.
- Any employee or departmentthat receives a service is a customer. Some common services provided to internalemployees include organizations known to be leaders in one or more aspects of their operations.

KEY WORDS

- **Cloud:** A technology that allows us to access our files and/or services through the internet from anywhere in the world. Technically speaking, it's a collection of computers with large storage capabilities that remotely serve requests.
- **Software:** A set of programs that tell a computer to perform a task. These instructions are compiled into a package that users can install and use. For example, Microsoft Office is an application software.
- **Domain:** a group of computers, printers and devices that are interconnected and governed as a whole. For example, your computer is usually part of a domain at your workplace.
- **Virtual Private Network (VPN):** A tool that allows the user to remain anonymous while using the internet by masking the location and encrypting traffic.
- **IP Address:** An internet version of a home address for your computer, which is identified when it communicates over a network; For example, connecting to the internet (a network of networks).

Notes

- **Exploit:** A malicious application or script that can be used to take advantage of a computer's vulnerability.
- **Breach:** The moment a hacker successfully exploits a vulnerability in a computer or device, and gains access to its files and network.
- **Firewall:** A defensive technology designed to keep the bad guys out. Firewalls can be hardware or software-based.
- **Malware "the bad guy":** An umbrella term that describes all forms of malicious software designed to wreak havoc on a computer. Common forms include: viruses, trojans, worms and ransomware.
- **Virus:** A type of malware aimed to corrupt, erase or modify information on a computer before spreading to others. However, in more recent years, viruses like Stuxnet have caused physical damage.
- **Ransomware:** A form of malware that deliberately prevents you from accessing files on your computer – holding your data hostage. It will typically encrypt files and request that a ransom be paid in order to have them decrypted or recovered. For example, WannaCry Ransomware. For more information on Ransomware, check out our free Ransomware Guide.
- **Trojan horse:** A piece of malware that often allows a hacker to gain remote access to a computer through a "back door".
- **Worm:** A piece of malware that can replicate itself in order to spread the infection to other connected computers.
- **Bot/Botnet:** A type of software application or script that performs tasks on command, allowing an attacker to take complete control remotely of an affected computer. A collection of these infected computers is known as a "botnet" and is controlled by the hacker or "bot-herder".
- **Spyware:** A type of malware that functions by spying on user activity without their knowledge. The capabilities include activity monitoring, collecting keystrokes, data harvesting (account information, logins, financial data), and more.
- **Rootkit:** Another kind of malware that allows cybercriminals to remotely control your computer. Rootkits are especially damaging because they are hard to detect, making it likely that this type of malware could live on your computer for a long time.
- **DDoS:** An acronym that stands for distributed denial of service – a form of cyber attack. This attack aims to make a service such as a website unusable by "flooding" it with malicious traffic or data from multiple sources (often botnets).
- **Phishing or Spear Phishing:** A technique used by hackers to obtain sensitive information. For example, using hand-crafted email messages designed to trick people into divulging personal or confidential data such as passwords and bank account information.
- **Encryption:** The process of encoding data to prevent theft by ensuring the data can only be accessed with a key.
- **BYOD (Bring Your Own Device):** Refers to a company security policy that allows for employees' personal devices to be used in business. A BYOD policy sets limitations

Notes

and restrictions on whether or not a personal phone or laptop can be connected over the corporate network.

- **Pen-testing:** Short for "penetration testing," this practice is a means of evaluating security using hacker tools and techniques with the aim of discovering vulnerabilities and evaluating security flaws.
- **Social Engineering:** A technique used to manipulate and deceive people to gain sensitive and private information. Scams based on social engineering are built around how people think and act. So, once a hacker understands what motivates a person's actions, they can usually retrieve exactly what they're looking for – like financial data and passwords.
- **Clickjacking:** A hacking attack that tricks victims into clicking on an unintended link or button, usually disguised as a harmless element.

REVIEW QUESTIONS

1. Explain the basics of information security.
2. What are the types of attacks? Explain
3. What is virus? How will you control it?
4. Briefly explain the role of hackers.
5. What is Cryptography? Explain.
6. How will you create a secure environment? Discuss.
7. What is Internet Security Standards? Briefly explain.
8. What is risk management? Discuss the types of risk.

FURTHER READINGS

1. Software Engineering, Bharat Bhushan Agarwal, Sumit Prakash Tayal, Firewall Media.
2. Softwaree Engineering, D. Sunder, University Science Press
3. Kaniclides, A., C. Kimble.(1995) A Development Framework for Executive Information Systems. Proceedings of GRONICS'95, Groningen, The Netherlands, February.
4. Kimble, C., (1997) Assessing the Relative Importance of Factors Affecting Information Systems Success, University of York Technical Report Series, YCS 283, Department of Computer Science, York, UK. 5.

SWAMI VIVEKANAND SUBHARTI UNIVERSITY

UGC Approved Meerut

Where Education is a Passion ...

ACCREDITED WITH GRADE A NAAC